JOURNAL OF COMPUTER AND SYSTEM SCIENCES 35, 365–400 (1987)

Algebraic Solutions to Recursion Schemes*

DAVID B. BENSON AND IRÈNE GUESSARIAN†

Computer Science Department, Washington State University, Pullman, Washington 99164-1210

Received July 16, 1986

The main problem in recursive scheme theory is determining how to solve a scheme and express its solution. Up to now this was always achieved by adding restrictive hypotheses either on the schemes themselves, or on the domains where they take their values, e.g., assuming the domains have a metric or an order structure and are complete with respect to this structure, or are iterative. Here we develop a strictly algebraic theory of recursion schemes with second-order substitutions. As it is strictly algebraic, the theory applies not only to all recursion schemes on trees, but also to recursion schemes on arbitrary algebras presented in the usual way by generators and relations. In particular, this gives a semantics for nondeterminism and for process algebras.

1. INTRODUCTION

The study of recursion schemes can be viewed from several standpoints:

—first-order substitutions, corresponding to iterative schemes, where trees, i.e., terms, are substituted for variables.

—second-order substitutions, corresponding to LISP-like recursive schemes, where trees are substituted for function symbols in trees.

—higher order substitutions, corresponding to higher order recursive schemes, where trees are substituted for functionals or higher order function symbols in trees.

All these notions may be extended to algebras other than the free algebras whose carriers consist only of sets of well-formed terms (or trees).

First-order substitution has an extensive theory using just algebraic techniques [17, 24, 26, 32] as well as an extensive theory using notions of order [4, 5, 42, 52, 57, 58] or metrics [16]. The theory of second-order substitutions so far developed requires either a notion of order or metric [9, 10, 50]. The theory of higher order substitutions has been shown to be reducible to the

* Research partially supported by National Science Foundation Grants MCS-8003433 and DCR-8402305.

† Current address: Université Paris VII, U.E.R. de Mathématiques et Informatique, Tour 55–56, 1er Etage, 2, Place Jussieu, Paris Cédex 05, 75251, France.

0022-0000/87 $3.00

theory of first-order substitution for higher type rational schemes, and order-theoretic studies can be found in [25, 27, 28, 30].

This paper develops a strictly algebraic theory of second-order substitutions. The theory includes both tree algebras and algebras obtained from congruence on trees. Second-order substitution schemes are characterized here as certain homomorphisms generalizing the usual fixed point semantics in which substitution schemes are viewed as functionals, or morphisms, operating on domains of continuous functions. As the theory is strictly algebraic, well-known categorical tools may be applied to obtain the desired solutions. Coequalizers correspond to the intuitive notion of solving a scheme by iteration. Pushouts correspond to the pasting together of solutions of all schemes. These colimits make possible the construction of algebras in which all schemes have solutions.

This direct solution method enables us to avoid the drawbacks of the usual methods: adding restrictive hypotheses on the domains, such as ordered, metric, or iterative domains, or adding restrictive hypotheses on the schemes, such as Greibach or contracting schemes. Often, these are ad hoc restrictions, which provide a solution for a very specific kind of scheme, without the possibility of generalizing to arbitrary schemes. For instance, iterative algebras [26, 32] are well suited for iterative schemes, but not at all for recursive or nondeterministic schemes. Moreover, from an aesthetic standpoint, the ad hoc solutions generally result in slightly weird structures, where, e.g., a chain $(a_1, a_2, ..., a_n, ...)$ has a least upper bound, while the chain $(b_1, a_2, ..., a_n, ...)$ will have no least upper bound, because the former happens to be the unwinding of a scheme, whereas the latter is not.

Let us discuss in more detail the least fixed point approach since it is the most widely used. The first obvious point is that, in some cases, one might need a grasp on other fixed points as well: this needs a lot of extra work in the usual framework, cf. [16], whereas it can be done at no extra cost with our method, since we get a hold on all solutions; note that we also take into account ordered algebras and least fixed points. A second problem with least fixed points is that they are well-suited neither for nondeterministic schemes, nor for communicating processes. For communicating processes, we do not know of any completely satisfactory answer yet, however, see Hennessy and Milner [36, 37], and Rounds [53]. We think our approach will extend quite naturally, cf. [13, 45]. So we will only discuss here the nondeterministic case, making some remarks about communicating processes only in Section 5.

One solution for nondeterminism, [9], has been to throw in a metric; then one naturally ends up with greatest fixpoints instead of least ones, and gets a nice semantics only for the case of Greibach schemes. Trying to stick more closely to the least fixed point approach, Lehmann and Smyth [43], Plotkin [51], Smyth [55], had to introduce power domains, which give satisfying solutions only when we have domains with a flat ordering to start with. Otherwise, as advocated by Lehmann [40], Boudol [18] had to keep track of the way a particular solution is computed: this leads to introducing morphisms and category theory, [1]. So, by working with categorical tools in the first place, we obtain solutions which

generalize straightforwardly to certain cases of nondeterminism. We simply introduce an extra operator called "**or**" and a congruence relation expressing its properties: commutativity, associativity, idempotence, and distributivity over other operations.

Another asset of our method is that it is also valid, without any extra work, for the many sorted case which is of great interest for abstract data type theory. Our method also generalizes very easily to the case, most important for practical purposes, where some relations and equations hold between the base functions and operations, the operations often called constructors or combinators. Most of the usual methods collapse as soon as one tries to introduce an equation on the base functions. For example, see [35] for a discussion of the iterative case. A trivial example may help convince the reader: assume the signature is $B = \{\wedge, t, f, \bot\}$, i.e., we have one binary operation $\wedge$, and three constants t, f, and $\bot$. Let the following equations hold: $x \wedge t = x$, $x \wedge f = f$, for all x. Because the intended meaning of the $\wedge$ operation is *and*, it is quite natural to require also that the equation $\bot \wedge x = \bot$ hold for all x; but then we end up with $\bot = \bot \wedge f = f$; thus we have to equate f and $\bot$, collapsing the originally intended structure. This is troublesome if we intend to consider any predicates afterwards. This shows that we cannot reasonably add an ordering and a bottom element to the boolean structure described by the signature $B' = \{\wedge, t, f\}$. Since our method is purely algebraic and does not attempt to add any extra structure to what is given, we easily extend our constructions to the case where equations hold between base function symbols in Section 5.

We will give a constructive and purely algebraic characterization of the free algebra in which all schemes have at least one solution. Roughly speaking, this free algebra is obtained by taking a suitable quotient which ensures that each homomorphism corresponding to a scheme has a fixpoint. Usually, there will be more than one fixed point (see Section 3): this stems from the fact that, since we exclude any order or metric, no continuity constraint can be forced upon the solutions. Hence we have to model all possible solutions in all possible algebras, and usually there will be more than one solution in an arbitrary algebra, see [16, 20]. We nevertheless give all cases a canonical solution, which corresponds to the least solution in case the algebras are ordered.

J. Meseguer suggested that the purely algebraic notion of substitution which we use here is an adequate model for the evaluation process of LISP. He also suggested that our approach treating nonfree algebras with relations in Section 5 provides a framework well suited to the study of abstract data types as initial algebras.

This paper is organized as follows: Section 2 gives the algebraic and categorical framework; Section 3 describes the basic construction for finding the canonical solution to a scheme; in Section 4 we construct the free complete algebra in which all schemes have solutions, sometimes called the algebraically closed algebra, and relate it to the order theoretic constructions. Section 5 generalizes these results to algebras presented by generators and relations. An Appendix contains reminders about category theory.

2. Notation and Monads

The notation for trees and tree substitutions follows [34]. The notation and terminology from category theory follows [39], with the addition of a notation for composition in lexical order: If $f: A \to B$ and $g: B \to C$ are morphisms of a category, $f \cdot g: A \to C$ denotes the composition

$$f \cdot g = A \xrightarrow{f} B \xrightarrow{g} C.$$

The dot is frequently elided so that fg denotes $f \cdot g$. Composition in the usual mathematical order is always denoted by $g \circ f$: $fg = f \cdot g = g \circ f$.

Let Σ be a possible infinite set consisting of finitary *operators*. We say Σ is a *signature* or that Σ is a *finitary signature* when we wish to emphasize that the rank of all operators in Σ is finite.

From [44], a *monad* $M = (M, \mu, \eta)$ in a category $\mathbf{C}$ consists of a functor $M: \mathbf{C} \to \mathbf{C}$ and two natural transformations

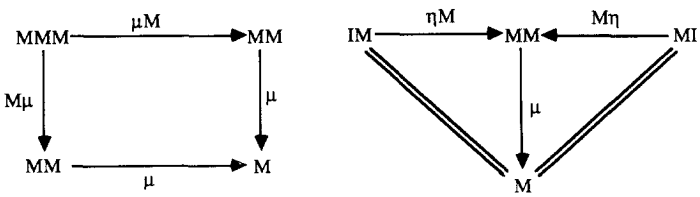
$$\mu: MM \dot{\to} M, \qquad \eta: I \dot{\to} M,$$

where I is the identity functor on $\mathbf{C}$; this data subject to the commutativity of Figs. 1 and 2.

Monads present exactly the same data as algebraic theories. Indeed, [33] calls monads "algebraic theories in monoid form." Arbib and Manes [7] give an introduction to the theory of monads.

Given a signature Σ and set A of so-called variables, let MA be the set of all trees generated by composition from the signature and the set of variables A. By varying over all sets A, one obtains a functor $M: \mathbf{Set} \to \mathbf{Set}$ which provides the set of all trees over variables A for each set A. Otherwise stated, MA is the carrier of the free M-magma over variables A. The natural transformation $\mu: MM \dot{\to} M$ is then the canonical reduction from trees over MA to trees over A, $\mu A: MMA \to MA$. Figure 1 says that trees over trees over trees over A may be canonically reduced to trees over A either top-down or bottom-up.

The natural transformation $\eta: I \dot{\to} M$ is the canonical insertion of the generators, sending each variable in A to a tree of height zero, $\eta A: A \to MA: a \mapsto \langle a \rangle$. Figure 2 guarantees that variables remain variables under the action of the tree reduction natural transformation μ.



Figures 1 and 2

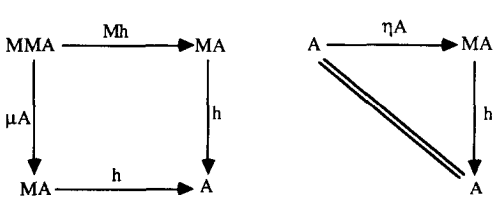
The above two paragraphs describe the construction of the free monad over a signature Σ. Since the signature is finitary, we may say that the monad is also finitary. As this is fixed throughout the paper, we now take *monad* to mean finitary monad although in general a monad M provides structures other than trees for each carrier A. In this paper, we will consider only the free monads in Sections 3 and 4 and monads which provide equivalence classes of trees as the structures over carriers in Section 5. As every monad may be obtained as the epi-image of a free monad in this way, in Section 5 we are in effect considering all monads over **Set**.

In this restricted setting, one may view a free finitary monad as the collection of sets of trees $\{MA \mid A$ is a set (of variables)$\}$ as in the example above. However, and as numerous authors have noted, the particular set of variables is of no direct interest. The use of monad notation suppresses this extraneous detail, resulting in a clear exposition of the essential facts. Further, the use of monad notation clarifies the distinction between the following three distinct types of homomorphisms: change of variables (part of the monad as functor), change of signature (a natural transformation which is a morphism of monads), and the structure map of an algebra (defined below).

Since there is quite enough to do simply for monads over **Set**, we do not discuss the generalization to arbitrary monads over arbitrary base categories. However, the exposition in terms of monads over **Set** makes it quite clear what must be done to generalize these results. If subsequent research directions in computer science suggest the value of similar results for a base category other than **Set**, these results may easily be obtained from this work. For example, all the results remain true for finitary monads over the base category **pSet** of pointed sets and point preserving functions, this category being isomorphic to the category of sets and partial functions. We shall not prove this here, under the assumption that remaining in the better understood framework of trees over **Set** is an aid to the intuition in what follows.

The *algebras* of the monad M are pairs (A, h), where A is an object in the base category **C** called the carrier of the algebra and an arrow $h: MA \to A$ of the base category called the *structure map* of the algebra, such that the diagrams in Figs. 3 and 4 commute. Figure 3 says that h may be evaluated by induction on structures while Fig. 4 says that elements of the carrier are already fully evaluated. Figure 3 may be viewed as

$$h(f(t_1, ..., t_n)) = f_A(h(t_1), ..., h(t_n))$$



FIGURES 3 AND 4

with the left side of the equation an instance of going down and right while the right side of the equation is an instance of going across and down.

The free M-magma over variables A, as an algebra of the monad M, is the algebra $(MA, \mu A)$. This is readily verified by noting that $\mu(MA) = (\mu M)A$, that is, the MA^{th} component of μ is identical to the A^{th} component of μM. The identification of the collection of free M-magmas with the monad M makes possible the treatment of schemes as monad morphisms.

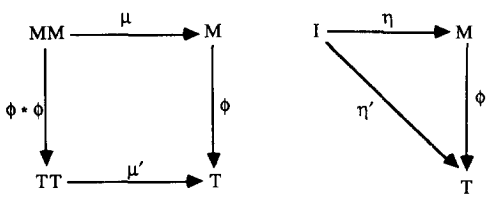
Let $M = (M, \mu, \eta)$ and $T = (T, \mu', \eta')$ be monads over base category **C**. A natural transformation $\phi: M \to T$ is a *monad morphism* if it preserves the structure, viz., see Figs. 5 and 6, where $\phi \star \phi = \phi M \cdot T\phi = M\phi \cdot \phi T$. A monad morphism $\phi: M \to M$ is a *monad endomorphism.*

The structure of a monad endomorphism ϕ for a free monad is particularly simple. For each set of variables V, $\phi(v) = v$ for each $v \in V$, while for each function symbol f in the signature let $\ddot{\phi}(f) \in MV$ be a tree of arity at most the rank of f. For each tree $f(t_1, ..., t_n) \in MV$, $\phi(f(t_1, ..., t_n)) = \ddot{\phi}(f)(\phi(t_1), ..., \phi(t_n))$. This is the substitution of $\ddot{\phi}(f)$ for f, for all f in F, sometimes called *full*, or *Kleene*, substitution. Figure 7 provides an illustration of this process. For further discussion of monad morphisms, see [5].

Each recursion scheme $s: G_i(x_1, ..., x_{n_i}) = t_i(x_1, ..., x_{n_i})$, $i \in I$, corresponds to a natural substitution of t_i for G_i; this translates into a homomorphism of free $F \cup \Phi$-algebras, i.e., into the monad morphism ϕ defined by: $\ddot{\phi}(f) = f$, for all f in F, $\ddot{\phi}(G_i) = t_i$, for all G_i in Φ. The details are given in Section 3.

If $\phi: M \to T$ is a monad morphism, it is *ipso facto* a homomorphism, for each set A, from the free algebra $(MA, \mu A)$ to the free algebra $(TA, \mu' A)$. As these are the free M-magma and the free T-magma, respectively, the monad morphism $\phi: M \to T$ endows the free T-magma with an M-magma structure by its action.

The category of finitary monads over **Set** and their morphisms is, by the equivalence to finitary algebraic theories, cocomplete, [56, Section 18.1.12]. In particular, coequalizers, multiple coequalizers, pushouts, and multiple pushouts always exist and these categorical constructions are used to find the monads in which recursion equations have solutions. The colimits are constructed "at the level of **Set**" [46], equivalently, by "pointwise evaluation," [39, Section 25.6]. Any base category **C** for which the category of finitary monads over **C** has coequalizers and multiple pushouts may replace **Set** in the following to obtain valid results, possibly less pleasant as Theorem 3.12 may not hold for **C**.



FIGURES 5 AND 6

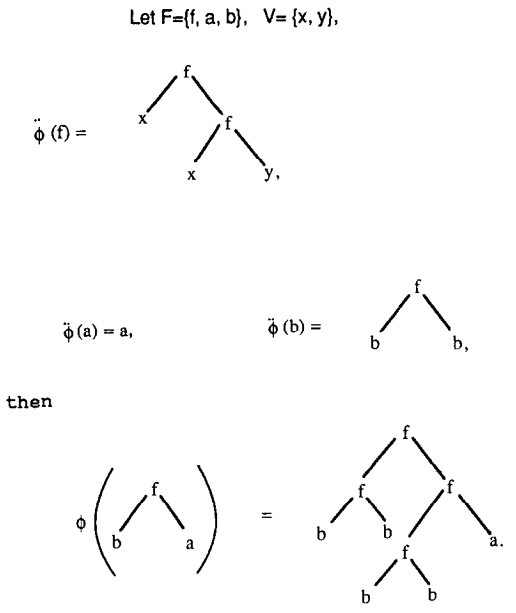


FIGURE 7

The first paper that uses monads to study the structure of tree transformations is [6]. Related topics may be found in Chapter 4 of [46]. For the reader's convenience, and in order to make the paper self-contained, the category theory notions we use are recalled in an Appendix.

3. The Basic Construction

In the present section, we show that coequalizers provide solutions for recursive schemes; these solutions correspond to the usual infinite unfoldings.

Let F (resp. Φ) be a ranked alphabet of finitary base function symbols (resp. function variables). The notation F_n (resp. Φ_n) denotes the function symbols (resp. function variables) of rank n. F and Φ may be infinite.

Let $M- = M(F, -)$ and $M_\Phi - = M(F \cup \Phi, -)$. That is, for any set of variable symbols V, MV is the free F-magma generated by V and $M_\Phi V$ is the free $F \cup \Phi$-magma generated by V. M and M_Φ are monads.

A *recursion scheme* is a set of ordered pairs indexed by some index set I. The first element of each ordered pair is a term $G_i(\mathbf{v})$ with $G_i \in \Phi$ and $\mathbf{v}$ a list from the set of variable symbols V. Considering only the first element of ordered pairs, each distinct ordered pair has a distinct function variable drawn from Φ. The corresponding

second element is a term $t_i(\mathbf{v}) \in M_\Phi V$ over the same *set* of variable symbols $\mathbf{v}$. These ordered pairs are called *recursion scheme equations* and are denoted by

$$G_i(\mathbf{v}) = t_i(\mathbf{v})$$

as is usual in recursion scheme theory. From the foregoing, each recursion scheme equation is an "equation" for a distinct function variable in Φ; no function variable participates in the left side of more than one recursion scheme equation within a single recursion scheme. A recursion scheme is then denoted by

$$s\colon G_i(\mathbf{v}) = t_i(\mathbf{v}), \qquad i \in I,$$

for some index set I. Thus s is a function from I to sets of ordered pairs. We take such functions to be injective in order to avoid redundant eqautions. We may abbreviate "recursion scheme" to just *scheme* and may write simply s to denote some scheme which is the current focus of attention.

Let $s\colon G_i(\mathbf{v}) = t_i(\mathbf{v})$, $i \in I$ be a recursion scheme, where $G_i \in \Phi$ and $t_i(\mathbf{v}) \in M_\Phi V$, $\mathbf{v}$ a list from V. For each V, the set of trees $M_\Phi V$ can be given two different $F \cup \Phi$-magma structures. First, the structure of the free algebra $(M_\Phi V, \mu_{M_\Phi V})$ defined by

$$\forall \theta \in F \cup \Phi, \qquad \forall \mathbf{t} \in \mathbf{M}_\Phi \mathbf{V}\colon \qquad \theta_{M_\Phi}(\mathbf{t}) = \theta(\mathbf{t}),$$

which is just the monad structure of M_Φ.

Second, the structure determined by the scheme $s\colon \forall \mathbf{t} \in \mathbf{M}_\Phi \mathbf{V}$,

$$\forall f \in F\colon \qquad f_{M_\Phi^s}(\mathbf{t}) = f(\mathbf{t}),$$
$$\forall G_i \in \Phi\colon \qquad G_{i,\, M_\Phi^s}(\mathbf{t}) = t_i(\mathbf{t}),$$

which is denoted M_Φ^s. Since $M_\Phi V$ is the free $F \cup \Phi$-magma, there is a unique morphism $\phi_s\colon M_\Phi \to M_\Phi^s$ which leaves M fixed and satisfies

$$\forall \mathbf{t}\colon \qquad \phi_s(G_{i,\, M_\Phi}(\mathbf{t})) = G_{i,\, M_\Phi^s}(\phi_s(\mathbf{t})).$$

In particular, this implies

$$\phi_s(t) = t \qquad \text{for any } t \text{ in } MV \qquad \text{and} \qquad \phi_s(v) = v \qquad \text{for any } v \text{ in } V$$

and, by induction on the depth of trees,

$$\phi_s(f(t'_1, \ldots, t'_n)) = f(\phi_s(t'_1), \ldots, \phi_s(t'_n)),$$
$$\phi_s(G_i(t'_1, \ldots, t'_{n_i})) = t_i(\phi_s(t'_1), \ldots, \phi_s(t'_{n_i}))$$

for all f in F, G_i in Φ, $t'_1, \ldots, t'_n$ in $M_\Phi V$. Note that the carrier of M_Φ^s is contained in the carrier of M_Φ, hence ϕ_s is a mapping $M_\Phi \to M_\Phi$ which leaves M fixed. Moreover, letting $\ddot{\phi}_s(f) = f$, for f in F, and $\ddot{\phi}(G_i) = t_i$ for G_i in Φ, the induction just

above shows that ϕ_s is a morphism $M_\Phi \to M_\Phi$, hence is an endomorphism of the monad M_Φ.

3.0. EXAMPLES. (a) Let $F = \{suc, or\}$, $\Phi = \{G\}$, and s: $G(x) = (x \text{ or } suc(G(x)))$. The morphism ϕ_s is the unique endomorphism of M_Φ leaving *suc* and *or* fixed and sending $G(t)$ to $(t \text{ or } suc(G(t)))$ for any t in M_Φ.

(b) $F = \{0, suc, \times\}$. One can obviously introduce the integers and an addition denoted "+." Let

$$s\colon G(x) = \sum_{i=0}^{n} a_i x^i.$$

Assume, moreover, the usual associativity and distributivity properties of $\times$ and $+$. We can then represent any tree t in M_Φ in the form

$$t = \sum_{j=0}^{p} b_j x^j.$$

Then

$$\phi_s(t) = \phi_s\left(\sum_{j=0}^{p} b_j x_j\right) = \sum_{i=0}^{n} a_i \left(\sum_{j=0}^{p} b_j x^j\right)$$

is the usual polynomial composition which substitutes polynomial t for variable x in polynomial G.

This substitution corresponds to the pattern of call expansions of Arbib and Manes [8, 47], to the Scott–Strachey least fixed point semantics [54], and also the algebraic semantics of [5, 34, 50].

3.1. PROPOSITION. *There is a bijection between the set of schemes with function variables Φ and the set of monad endomorphisms of M_Φ.*

Proof. The above remarks show how to construct, for each scheme s, an endomorphism ϕ_s of M_Φ. Conversely, let ϕ be an endomorphism of Φ; then by the endomorphism property of ϕ, for any θ in $F \cup \Phi$, $\phi(\theta(\mathbf{t})) = \ddot{\theta}(\phi(\mathbf{t}))$ for any vector $\mathbf{t}$ in $M_\Phi V$, where $\ddot{\theta} \in M_\Phi V$ is defined by: $\ddot{\theta}(\mathbf{v}) = \phi(\theta(\mathbf{v}))$. Let F_ϕ be the set of symbols invariant under ϕ, i.e.,

$$F_\phi = \{\theta \mid \ddot{\theta} = \theta\} = \{\theta \mid \phi(\theta(\mathbf{v})) = \theta(\mathbf{v})\}.$$

Let s_ϕ be defined by: for G in $(F \cup \Phi) - F_\phi$: $G(\mathbf{v})) = \phi(G(\mathbf{v}))$. Clearly, s_ϕ is a recursive scheme over F, and $\phi_{s_\phi} = \phi$. ∎

The bijection described in Proposition 3.1 shall be taken for granted without any further details in the sequel; in particular, whenever we consider only one scheme, s,

we assume that $F = F_{\phi_s}$, unless otherwise stated, and we say that s is a scheme with function variables Φ.

3.2. *Remark.* Recall that a variable G_i in a scheme s is said to be *looping* if it contains a cycle in its definition, namely equations: $G_i(\mathbf{x}) = G_{i_1}(\mathbf{t}_1), \ldots, G_{i_p}(\mathbf{x}) = G_i(\mathbf{t}_p)$. In case F_ϕ is empty, s_ϕ is a recursive scheme involving only function variables. This models some relations holding between defined function variables. In abstract data type theory, such a set of equations could model some relations holding between the functions of Φ, considered as defined operations. From the standard program scheme theory standpoint though, such looping schemes are excluded. We see no reason to exclude them a priori.

We now define solutions of recursion schemes.

3.3. DEFINITION. Let (Y, y) be an algebra of the monad M_Φ, equivalently an $F \cup \Phi$-algebra. A *solution* of scheme s in (Y, y) is an assignment to each G_i of rank n_i in Φ of some γ_i: $Y^{n_i} \to Y$ such that: (i) γ_i belongs to the clone of (Y, y), namely the closed set generated by the base operations F, [19], (ii) the γ_i's satisfy the equations defining scheme s. A solution is said to be *canonical* iff $\gamma_i = y(G_i)$ for all i. Compare to [47, Chap. 7].

The difference between canonical and noncanonical solutions is illustrated in examples 3.13–3.15 below.

3.4. PROPOSITION. *There exists a canonical solution of scheme s in (Y, y) iff Fig. 8 is commutative.*

Proof. Assume Fig. 8 is commutative and let $\gamma_i = y(G_i)$. Then $y(G_i) = y(\phi(G_i)) = y(t_i)$, which implies that the γ_i's define a canonical solution. For the converse direction, note that y allows us to carry onto Y the two $F \cup \Phi$-algebra structures M_Φ and M_Φ^s of $M_\Phi Y$; the first one corresponds to the morphism y: $M_\Phi Y \to Y$ and the second one to the morphism $\phi Y \cdot y$: $M_\Phi Y \to Y$, which differ only on the G_i's. To say that the γ_i's define a canonical solution is the same as to say that the images of G_i's under these two morphisms coincide; then Fig. 8 commutes. ∎

Intuitively, the commutativity of Fig. 8 amounts to saying that the canonical solution is a fixed point of the equations corresponding to s in Y. In other words, Fig. 8 translates into: $\phi Y \cdot y = y = \mathrm{id} \cdot y$, where id is the identity on $M_\Phi Y$. The structure map y is thus a pre-coequalizer of the pair $(\phi Y, \mathrm{id})$, i.e., satisfies $\phi Y \cdot y = \mathrm{id} \cdot y$.

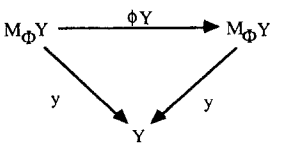


FIGURE 8

Intuitively then, the most general solution of s will be obtained by taking the initial object among the pre-coequalizers, namely the coequalizer of $(\phi Y, \mathrm{id})$.

We may also say that the algebra (Y, y) provides a fixed point of scheme s. Fixed point solutions may be obtained by iteration as in Fig. 9.

In other words, iterating by full substitution ϕ, as in the usual order theoretic framework, results in unfolding the scheme s and the unfoldments yield the same function under the structure map y.

3.5. *Remark.* In the standard order-theoretic framework, an interpretation is an order-complete F-algebra I; each scheme s has a least fixpoint in I which is an n-tuple $(G_1, ..., G_{n_I})$ of continuous functions $G_i: D_I^{n_i} \to D_I$, where D_I is the domain of I. See [34]. Two schemes s and s' are said to be *equivalent* iff, for all I, $G_{1_I} = G'_{1_I}$.

As in the usual fixed point theory, an *interpretation* is thus an $F \cup \Phi$-algebra (Y, y) satisfying: $y(G_{i, M_\Phi^s}) = y(G_{i, M_\Phi})$, for all G_i in Φ, or, equivalently, $y(t_i(\mathbf{x})) = y(G_i(\mathbf{x}))$, for all lists $\mathbf{x}$ in Y, where $G_i(\mathbf{x}) = t_i$ is the equation defining G_i. All typical domains can then be captured without adding an ordering. Moreover, since this theory generalizes equally well to heterogeneous algebras, more general interpretations which cannot be modelled via the flat ordering trick can be captured here, e.g., nondeterminism and algebras with relations between base function symbols. We return to this point in Sections 4 and 5.

Each (Y, y) making Fig. 8 commute will factor uniquely through the coequalizer of $\phi: M_\Phi Y \to M_\Phi Y$ and $\mathrm{id}: M_\Phi Y \to M_\Phi Y$, which then lifts to the level of monads as ϕ is a monad endomorphism. Therefore, let $(k, K) \approx \mathrm{Coeq}(\phi, \mathrm{id})$ be the coequalizer of $\phi: M_\Phi \to M_\Phi$ and $\mathrm{id}: M_\Phi \to M_\Phi$, where K is a monad with monad multiplication μ_K and $k: M_\Phi \to K$ is a monad morphism. At the level of sets, KY is a collection of congruence classes induced by $\phi(t) \equiv t$ for all $t \in M_\Phi Y$ and k is the morphism sending each t to its congruence class. Equivalently, KY is the set of congruence classes of the congruence generated from the scheme $G_i(\mathbf{x}) = t_i$, $i \in I$ by $G_i(\mathbf{x}) \equiv t_i$, $i \in I$. This congruence will be denoted by $\equiv_\phi$ or just by $\equiv$ when ϕ is understood from the context.

3.6. PROPOSITION. *For each scheme s and set Y, $(k, K) \approx \mathrm{Coeq}(\phi_s, id)$ determines an algebra C_Y in which scheme s has a canonical solution.*

Proof. Let C_Y be the structure map $M_\Phi KY \to^{kKY} KKY \to^{\mu_k Y} KY$ together with the carrier KY. C_Y is an algebra of the monad M_Φ as may be determined by check-

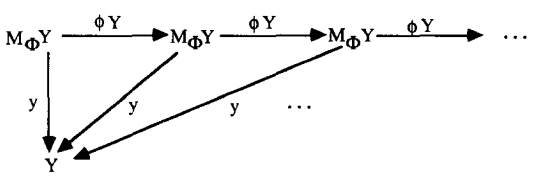


FIGURE 9

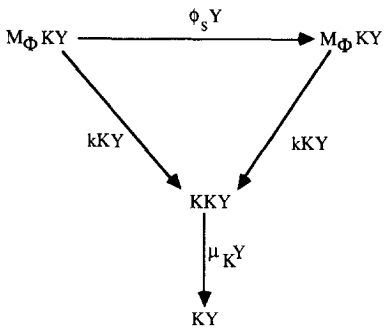


FIGURE 10

ing the details of Figs. 3 and 4. By the coequalizer property, Fig. 10 commutes. By Proposition 3.4, the algebra C_Y then has a canonical solution for s. ▮

3.7. COROLLARY. *The algebra $C_\varnothing$ has the following initial property: for every algebra (Y, y) having a canonical solution for s, there exists a unique morphism $\hat{y}: C_\varnothing \to (Y, y)$ of $F \cup \Phi$-algebras rendering commutative Fig.* 11, *where i is the canonical injection $i: M_\Phi \varnothing \to M_\Phi Y$ and $(k\varnothing, K\varnothing) \approx \mathrm{Coeq}(\phi_s \varnothing, \mathrm{id}\varnothing)$.*

Proof. Since (Y, y) has a canonical solution, all the diagrams in Fig. 12 commute; hence $\phi_s iy = iy$ and by the coequalizer property of $k\varnothing$, there exists a unique $\hat{y}$ making Fig. 11 commute. A tedious verification shows that $\hat{y}$ is indeed a morphism of $F \cup \Phi$-algebras. ▮

With this background we may now suppress the explicit mention of carriers or sets of variables. Thus we may write expressions such as $t \in M$ to denote a tree, sometimes called a term, t, which is an M-structure over some carrier. Similarly, we may write an expression $t \in M_\Phi$ to denote a tree which is an M_Φ-structure with respect to an unstated carrier.

3.8. DEFINITION. For scheme s, let $t \to^s t'$ iff there exist $u, v, \mathbf{w}$ such that $t = u[x \leftarrow v(\mathbf{w})]$ and $t' = u[x \leftarrow \phi(v)(\mathbf{w})]$, where x is a new variable symbol having only one occurrence in u, while u, v are trees in M_Φ, $\mathbf{w}$ is an n-tuple of terms in M_Φ, and $u[x \leftarrow t]$ denotes the substitution of t for x in u. Define $t \leftrightarrow^s t'$ iff $t \to^s t'$ or $t' \to^s t$.

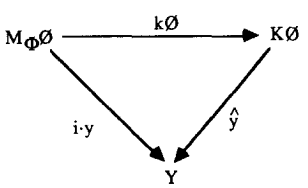


FIGURE 11

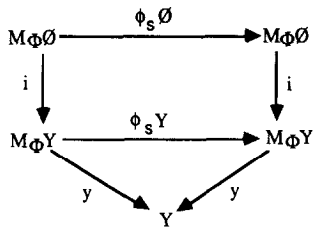


FIGURE 12

3.9. LEMMA. *For $\phi = \phi_s$, $t \equiv_\phi t'$ iff there exists a sequence $t_0, ..., t_n$ such that $t = t_0 \leftrightarrow^s t_1 \leftrightarrow^s t_2 \cdots \leftrightarrow^s t_n = t'$.*

3.10. LEMMA. *Let $t = u[x \leftarrow v(\mathbf{w})]$, using the notation of Definition* 3.8. *Assume $\phi^n(t) \in M$. Then*

(i) $\phi^n(u) \in M$,

(ii) *if x occurs in $\phi^n(u)$, then $\phi^n(v) \in M$.*

Proof. As ϕ is a monad endomorphism on the free finitary monad M_Φ, $\phi^n(t) = \phi^n(u)[x \leftarrow \phi^n(v)(\phi^n(\mathbf{w}))]$. Therefore every function symbol of $\phi^n(u)$ occurs in $\phi^n(t)$; hence if all function symbols in $\phi^n(t)$ are base function symbols, so are all function symbols in $\phi^n(u)$. The same holds for $\phi^n(v)$ if x occurs in $\phi^n(u)$. ∎

3.11. LEMMA. *The following statements hold:*

(i) *if $t \rightarrow^s t'$ and $\phi^n(t) = b \in M$, then $\phi^n(t') = b$,*

(ii) *if $t \rightarrow^s t'$ and $\phi^n(t') = b \in M$, then $\phi^{n+1}(t) = b$.*

Proof. For (i), let $t = u[x \leftarrow v(\mathbf{w})]$ and let $t' = u[x \leftarrow \phi(v)(\mathbf{w})]$. There are two possibilities. Either x does not occur in $\phi^n(u)$ and then $b = \phi^n(t) = \phi^n(u) = \phi^n(t')$, or x occurs in $\phi^n(u)$ in which case $\phi^n(t) = \phi^n(u)[x \leftarrow \phi^n(v)(\phi^n(\mathbf{w}))]$. By Lemma 3.10, $\phi^n(u)$, $\phi^n(v)$ are in M. Therefore $\phi^{n+1}(v) = \phi^n(v)$ and $\phi^n(t') = \phi^n(t) = b$. The proof of (ii) is similar. ∎

3.12. THEOREM. *Let s be a scheme, $\phi = \phi_s$, and $(k, K) \approx \mathrm{Coeq}(\phi, \mathrm{id})$. $M \rightarrow^i M_\Phi \rightarrow^k K$ is injective.*

Proof. Recall that $i: M \rightarrow M_\Phi$ is the injection of M into M_Φ. We prove that each congruence class of $\equiv_\phi$ contains at most one element of M. Suppose $b \in M$ and $b \equiv_\phi t$. By Lemma 3.9, for some n we have $b = t_0 \leftrightarrow^s t_1 \leftrightarrow^s t_2 \leftrightarrow^s \cdots \leftrightarrow^s t_n = t$. By repeated application of Lemma 3.11, and remarking that $\phi(b) = b$, we have $\phi^m(t) = b$ for some $m \leqslant n$. If $t \in M$, then $\phi(t) = t$, hence $t = \phi^m(t) = b$. ∎

This theorem guarantees that the morphism k to the coequalizing monad K introduces no identifications of M-structures. In particular, no two function symbols of F are identified. In Section 5, we will write $\psi = i \cdot k$ for this injection of the base monad into the coequalizing monad.

We conclude this section with several examples of solutions and canonical solutions in the coequalizing monad.

3.13. EXAMPLE. Solutions to scheme s can be found in a canonical way in K. Usually there will be more than one solution in K, although we always have a single canonical solution. Let s be $G(x)=g(G(x))$ with $F=\{g, a\}$. The ranks of elements of F are $r(g)=1$ and $r(a)=0$. Then $M_\Phi\varnothing=\{(g^*G^*)^*(a)\}$, and $K\varnothing=M_\Phi\varnothing/\equiv$ can be depicted as in Fig. 13.

Clearly, in $K\varnothing$, the canonical solution to scheme s is the function G, but all the functions $G^k\circ g^n$, for $n\geq 0$ and $k>0$ in N, are also solutions to s. Hence s has an infinity of solutions in $K\varnothing$.

Let us check that K, with the obvious morphism, is the coequalizer of (ϕ_s, id). Now k clearly satisfies $\phi\cdot k=k$. Since k is a regular epi, for any $k'\colon M_\Phi\to K'$ there will exist at most one $j\colon K\to K'$ such that $k\cdot j=k'$, since $k\cdot j=k\cdot j'$ and k epi imply $j=j'$. So let $k'\colon M\to K'$ be such that $\phi k'=k'$, and let us construct j such that $k'=k\cdot j$. The map j is pointwise defined by $j(a)=k'(a)$, and for $n, l\in N$,

$$
\begin{aligned}
&j((g^*G)^n(g^l(a)))\\
&\quad=k'((g^*G)^n(g^l(a)))=k'(G^n(g^l(a))), \qquad \text{since } \phi\cdot k'=k',\\
&\quad=k'(G)^n(k'(g)(k'(a))), \qquad \text{since } k' \text{ is a morphism.}
\end{aligned}
$$

Let us check that j is a morphism: Let $t=(g^*G)^n(g^l(a))$. Then

$$
\begin{aligned}
&j(G(t))=j(G^{n+1}(g^l(a)))=k'(G)^{n+1}k'(g)^l(a)\\
&\quad=k'(G)(j(t))=j(G)\,j(t);\\
&j(g(t))=k'(g)^{l+1}(k'(a))=j(g)\,j(t) \qquad \text{if } n=0,\\
&j(g(t))=j(t)=k'(G)(k'(G^{n-1})(k'(g)^l(k'(a))))\\
&\quad=k'(G)(t')=\phi\cdot k'(G)(t'), \qquad \text{since } \phi\cdot k'=k',\\
&\quad=k'(g\circ G)(t')\\
&\quad=k'(g)(k'(G)(t'))=k'(g)(j(t)), \qquad \text{since } k' \text{ is a morphism,}\\
&\quad=j(g)(j(t)) \qquad \text{if } n>0.
\end{aligned}
$$

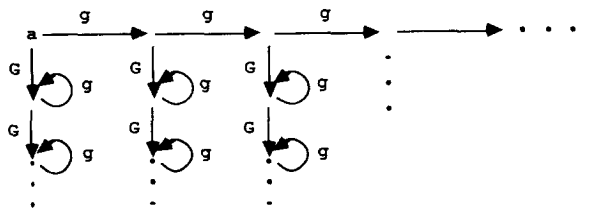


FIGURE 13

3.14. EXAMPLE. Even a nontrivial Greibach scheme may have several solutions in K, as shown by the following example, due to B. Courcelle. Let scheme s be defined by:

$$s: \begin{cases} G(x) = f(x, G(G(x))) \\ H(x) = f(G(x), H(G(x))) \end{cases}$$

$$F = \{f, a\} \qquad \text{and} \qquad r(f) = 2,\ r(a) = 0.$$

Then, in $K\varnothing$, the pair (G, G^2) is a solution of s which is different from the canonical solution (G, H).

3.15. EXAMPLES. The specificity of canonical solutions is also well illustrated by a slight variation of Example 3.13. Suppose A is an order-complete algebra, where scheme s has a least solution. Let $\gamma: K\varnothing \to A$ be a morphism taking the canonical solution in $K\varnothing$ to the least solution in A. Then γ does not necessarily take other solutions to s in $K\varnothing$ to the least fixpoint in A, unless s is a Greibach scheme which of course has a unique solution in A. Let $F = \{\Omega, a, g\}$, with $r(a) = r(\Omega) = 0$, $r(g) = 1$, s: $G(x) = G(g(x))$. $K\varnothing$ and $A = A(F)$ are depicted in Fig. 14, where $A(F)$ is the free complete ordered algebra generated by F; its ordering is depicted by upgoing arrows. The solutions in A are all the constant functions; the least solution, or least fixed point, is the constant function $y(x) = \Omega$ for all x. The solutions in $K\varnothing$ are all the functions $g^*G^+(g^*G^*)^*$, together with all constant functions. The canonical solution is G.

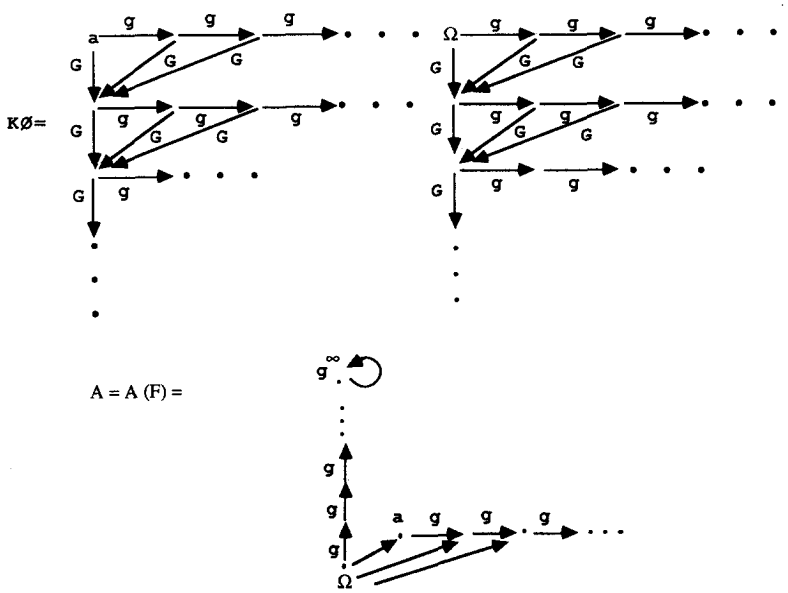


FIGURE 14

Let γ send the canonical solution G into the least fixpoint Ω in A. Let ψ_n be the following solution to s in $K\varnothing$: $\psi_n(G)(x) = g^n(\Omega)$ for all x. Then γ sends ψ_n to the constant function $c_n(x) = g_n(\Omega)$ for all x, which is a solution of s in A different from the least one. Notice that all solutions in A, except g^∞, can be obtained this way. The solution g^∞ can only be obtained via the initial property of $K\varnothing$ and the morphism μ sending the canonical solution to the constant function g^∞, $\mu(G) = g^\infty$. Then nonconstant solutions in $K\varnothing$ are mapped by γ into constant solutions of s in A the following way:

$$\forall x\colon \gamma(g^{n_1}G^{p_1} \cdots g^{n_r}G^{p_r})(x) = \begin{cases} \Omega & \text{if } p_r \neq 0, \\ g^{n_r}(\Omega) & \text{if } p_r = 0. \end{cases}$$

4. Construction of the Algebraically Closed or Complete Magma

In the previous section, we constructed for a given scheme s a monad K_s in which s had a canonical solution. The construction can be schematized as in Fig. 15, where i is the injection $M \to M_{\Phi_s}$ and $(k_s, K_s) \approx \mathrm{Coeq}(\phi_s, \mathrm{id})$.

We will now construct a monad in which all schemes on $F \cup \Phi$ have a solution and which is minimal in the sense that it consists only of solutions of recursive schemes. The idea is rather simple in essence. It consists in constructing a monad which is closed with respect to the property: schemes have solutions. This closure monad is obtained by pasting together, via a multiple pushout, all canonical solutions to recursion schemes. A similar approach can be found in [31] but it presupposes an ordering on the domains. Namely, order the domains, then complete them by adding all the least upper bounds which are necessary for schemes to have a solution. Thus one adds all unwindings of schemes. This results in a slightly unnatural notion of order-completeness [3, 31, 33, 22]. Moreover, the addition of an ordering can result in weird collapsing if the operations satisfy additional equations. Suppose, e.g., $F = \{e, {}^{-1}, \cdot\}$, with $r(e) = 0$, $r({}^{-1}) = 1$, $r(\cdot) = 2$, together with the axioms which make an F-algebra into a group with unit e. Then, throwing in an extra $\perp$ element results in: $\perp = e \cdot \perp = e \cdot \perp \cdot \perp^{-1} = e$. Similarly, assuming the strictness of "$\cdot$", $x \cdot \perp = \perp$, leads to $x = \perp$ for all x, hence the only strict ordered group is the trivial group.

So here we avoid adding an ordering and just add in all solutions to recursion schemes. This directly results in a recursive completion without resorting to an extra order-completion. Moreover, our method will be directly generalizable to the case where equations hold between base function symbols in Section 5.

In order to stay within the program scheme formalism, we will suppose that all the schemes we consider have a fixed set F of base function symbols which is left

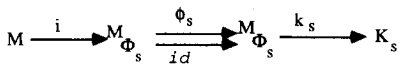


FIGURE 15

unchanged by all the ϕ_s. This hypothesis is, however, superfluous as will appear in the statement of our results: It suffices in fact to define $F = \bigcap_s F_{\phi_s}$.

4.1. DEFINITION. Two schemes s and s' are isomorphic if they are interconvertible by a permutation of the equations or a renaming of the function variables. Let S be a system of representatives of the isomorphism classes of schemes.

4.2. DEFINITION. Let P be the pushout of $\{i \cdot k_s \mid s \in S\}$.

We may assume that any two schemes in S have, by renaming if necessary, disjoint sets of function variables. The algebraic closure P of M is the pushout of all the K_s for $s \in S$. The assumption of disjoint function variables implies that the canonical solutions in P of different schemes will be different.

From this assumption, one may alternatively construct P as the multiple coequalizer of the set $\{\zeta_s : M \to M_\Phi \mid s \in S\}$, where $\zeta_s = M \to^i M_{\Phi_s} \to^{i^\nabla} M_\Phi \to^{\phi_s^\nabla} M_\Phi$ with Φ the alphabet of all function variables, i^∇ the natural injection and ϕ_s^∇ equal to ϕ_s on M_{Φ_s} while ϕ_s^∇ is the identity on $M_{\Phi - \Phi_s}$. Obviously, P, together with the pushout (or coequalizer) mapping corresponding to s, provides algebras of M_Φ. Now, by construction, every scheme s has a solution in P. We will see in Corollary 4.6, that, in addition, any new scheme we might define on $F \cup \Phi$ also has a solution obtained by pasting solutions together. Alternatively and intuitively, the construction of P can also be viewed as a classical completion construct: for instance, it is similar to the process of completing the rationals to an algebraically closed field, the difference being that the present completion construct stops after just one step of the usual inductive completion, because an algebraically closed structure is reached after just one completion step. Hence the following theorem.

4.3. THEOREM. *P is algebraically closed, i.e., any scheme on $F \cup \Phi$ has a canonical solution in P.*

The proof will proceed in several lemmas.

4.4. LEMMA. *Let $F, \Phi_j, j \in J$, be a set of base function symbols and an indexed collection of sets of variable function symbols and let $\Phi = \bigsqcup_{j \in J} \Phi_j$ denote the disjoint union of the Φ_j's. Let $i_j : M \to M_{\Phi_j}$, $\bar{i}_j : M_{\Phi_j} \to M_\Phi$ be canonical injections. Figure* 16 *is a pushout for all $j, l \in J$.*

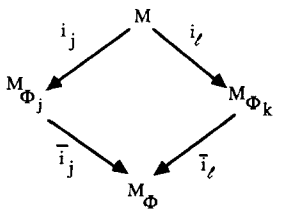


FIGURE 16

Proof. Clearly $i_j \cdot \bar{i}_j = i_l \cdot \bar{i}_l$ for all $j, l \in J$. Let $(\{f_j\}_J, A)$ be such that $i_j \cdot f_j = i_l \cdot f_l$ for all $j, l \in J$. Then, by the monad morphism definition, A has a Φ_j-magma structure deduced from the one of M_{Φ_j} and f_j is a magma-homomorphism for $j \in J$. Hence A has a Φ-magma structure, and by the universal property of M_Φ there exists a unique Φ-magma homomorphism $\bar{f}: M_\Phi \to A$ such that $\bar{i}_j \cdot \bar{f} = f_j$, $j \in J$. ∎

The above lemma is a restatement for free monads of the following fact: in **Set** the pushout of $(\varnothing, \{\varnothing \to \Phi_j\}_J)$ is $(\{in_j: \Phi_j \to \Phi\}_J, \Phi = \bigsqcup_{j \in J} \Phi_j)$ with in_j the canonical injections into the disjoint union.

The *disjoint union* of schemes s_1 and s_2 is denoted $s_1 \sqcup s_2$. The two schemes are made disjoint by introducing new function variables as necessary: If s_i is a recursion scheme over function variables Φ_i and indexed by I_i, $i = 1, 2$, the disjoint union scheme $s_1 \sqcup s_2$ is a scheme over function variables $\Phi_1 \sqcup \Phi_2$ and indexed by $I_1 \sqcup I_2$. In this way one may always determine of each recursion scheme equation in $s_1 \sqcup s_2$ from which scheme, s_1 or s_2 it arose. Further, the recursion scheme equations in s_i cannot influence solutions for scheme s_{3-i}, $i = 1, 2$, within $s_1 \sqcup s_2$. The generalization to the disjoint union of an indexed family of schemes s_j, $j \in J$, is denoted $\bigsqcup_{j \in J} s_j$ and is a scheme over function variables $\bigsqcup_{j \in J} \Phi_j$, where Φ_j is the set of function variables for scheme s_j, etc.

4.5. LEMMA. *Let* $(k_j, K_j) \approx \mathrm{Coeq}(\phi_j, \mathrm{id})$ *for* $j \in J$ *and* $(k_J, K_J) \approx \mathrm{Coeq}(\phi_J, \mathrm{id})$, *where* ϕ_j, ϕ_J *are determined by the schemes* s_j, $\bigsqcup_{j \in J} s_j$, *respectively. Then* K_J *is isomorphic to* P, *the pushout of* $(M, \{i_j \cdot k_j\}_J)$.

Proof. Let $J = \{1, 2\}$ and write ϕ_{12} for ϕ_J, k_{12} for k_J, etc., to consider Fig. 17, where (χ_1, χ_2, P) is the pushout of $(M, i_1 \cdot k_1, i_2 \cdot k_2)$, $(k_{12}, K_{12}) \approx \mathrm{Coeq}(\phi_{12}, \mathrm{id})$, the $\hat{k}_j$ for $j = 1, 2$ are uniquely determined by $\bar{i}_j k_{12} = k_j \hat{k}_j$, and $\bar{m}$ is uniquely determined by the equations $\hat{k}_j = \chi_j \cdot \bar{m}$, $j = 1, 2$. Calculating, $i_1 \phi_1 \bar{i}_1 = i_1 \bar{i}_1 = i_2 \bar{i}_2 = i_2 \phi_2 \bar{i}_2$ so there exists a unique ϕ such that $\phi_j \bar{i}_j = \bar{i}_j \phi$. That $\phi = \phi_{12}$, the endomorphism determined by the scheme $s_1 \sqcup s_2$, follows by noting that $\phi_j \bar{i}_j = \bar{i}_j \phi_{12}$, $j = 1, 2$.

Continuing the calculations, $i_1 \phi_1 k_1 p_1 = i_2 \phi_2 k_2 p_2$ so there exists a unique $t: M_\phi \to P$ such that $\bar{i}_j \cdot t = \phi_j k_j \chi_j = k_j \chi_j$ and then $\bar{i}_j \phi_{12} t = \phi_j \bar{i}_j t = \phi_j k_j \chi_j = k_j \chi_j = \bar{i}_j t$, $j = 1, 2$. As $(\{\bar{i}_j\}, M_\Phi)$ is a colimit, it is an epi-sink, hence $\phi_{12} \cdot t = t$. Therefore there exists a unique $t': K_{12} \to P$ such that $t = k_{12} \cdot t'$ so $k_j \cdot \chi_j = \bar{i}_j \cdot k_{12} \cdot t' = k_j \cdot \hat{k}_j \cdot t'$ and $\hat{k}_j = \chi_j \cdot \bar{m}$, whence $k_j \cdot \chi_j = k_j \cdot \chi_j \cdot \bar{m} \cdot t'$, $j = 1, 2$. But k_j is regular epi, so $\chi_j = \chi_j \cdot \bar{m} \cdot t'$, $j = 1, 2$. Now $(\{\chi_j\}, P)$, as a colimit, is an epi-sink, so $\bar{m} \cdot t' = \mathrm{id}_P$.

For the other direction, $\bar{i}_j \cdot k_{12} = k_j \cdot \hat{k}_j = k_j \cdot \bar{m} = \bar{i}_j \cdot t \cdot \bar{m} = \bar{i}_j \cdot k_{12} \cdot t' \cdot \bar{m}$, $j = 1, 2$. As $\{\bar{i}_j\}$ is an epi-sink, $k_{12} = k_{12} \cdot t' \cdot \bar{m}$. Now k_{12} is regular epi, so $\mathrm{id} = t' \bar{m}$ and t', $\bar{m}$ are isomorphisms.

The generalization to arbitrary J is straightforward and rather tedious. ∎

4.6. COROLLARY. *Any scheme* s *on* P (*defined by an endomorphism of* P *leaving* M *fixed*) *has a solution in* P.

Proof. Let $s: G(\mathbf{v}) = t_i$, $i \in I$, be a scheme on P. Let t_j, $j \in J$ be the elements of

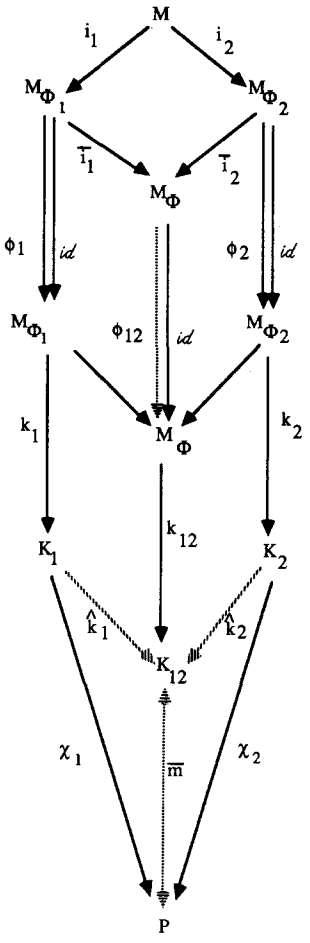


FIGURE 17

P occuring in the t_i. Each t_j is the solution of some scheme s_j on M_{Φ_j}. Let s' be a representative of the isomorphism class of $s \sqcup \bigsqcup_{j \in J} s_j$ on $M_{\Phi'}$ where $\Phi' = \Phi \sqcup \bigsqcup_{j \in J} \Phi_j$. Then by Lemma 4.5 (which holds for any pushout), $K_{s'}$ is isomorphic to the pushout of $(M, i \cdot k_s, \{i_j \cdot k_{s_j}\}_J)$. Now s' has a canonical solution in $K_{s'}$, hence so does s. Intuitively, the solution of s is obtained by combining the solutions of $s, s_j, j \in J$. The solution of s in P is obtained by embedding $K_{s'}$ in P and taking the image of the solution of s' in $K_{s'}$. ∎

This concludes the proof of Theorem 4.3.

The construction of P consists in adding solutions for recursive schemes, hence it can be viewed as an algebraic completion process, or an algebraic closure process, similar to the construction of the algebraic closure of a field.

A very nice feature of our construction is that it gives in a single step completion process an algebra which has solutions to all schemes and consists only of solutions of schemes. The same kind of construction will also be shown to be valid for algebras with relations in the next section. On the other hand, the usual order or metric based completion processes, while giving smaller completions, require transfinite induction [2, 11, 22, 30, 33, 48, 49].

4.7. DEFINITION. Let $\mathrm{Sch}(F)$ be the set of finite and infinite trees over F which are the solutions of some program scheme.

$\mathrm{Sch}(F)$ is the same as RCT_F of [31]. Actually, the basic idea of the construction of $\mathrm{Sch}(F)$ is the above-mentioned completion or closure process. Similar ideas are well known in rational languages and formal power series theory and have been around for quite a while. For instance, Proposition 4.9 is first hinted at in [50] and is more explicitly investigated in [18]. For completeness sake, we gave the basic idea of the proof of Proposition 4.9 which is algebraic in essence and based on the idea of substitution. The next fact is well-known.

4.8. PROPOSITION. *If an infinite tree is in* $\mathrm{Sch}(F)$, *then its path language is a deterministic, prefix-free, context-free language* [21, 29].

4.9. PROPOSITION. $\mathrm{Sch}(F)$ *is algebraically closed.*

Proof. The same as the proof of Corollary 4.6. See also [50]. Briefly, let $s\colon G_i(V) = t_i$, $t_i \in \mathrm{Sch}(F \cup \Phi)$, $i \in I$. For $i \in I$ each t_i is in $\mathrm{Sch}(F \cup \Phi)$ hence is the solution of some scheme s on $F \cup \Phi$, having a set Φ_i of function variables. Without loss of generality we may assume that Φ and Φ_i are pairwise disjoint. Let $s' = s \sqcup \bigsqcup_{i \in I} s_i$ be the scheme over function constants F with function variables $\Phi \sqcup \bigsqcup_{i \in I} \Phi_i$, obtained by putting together the equations of s and of all the $\{s_i\}$. The solution of s is also the solution of s', hence belongs to $\mathrm{Sch}(F)$. ∎

4.10. PROPOSITION. $\mathrm{Sch}(F)$ *is a monad.*

Proof. Let $S = \mathrm{Sch}(F)$ so that SV is a set of trees over variables V. Let $t(\mathbf{v}), t_1, ..., t_n \in SV$ so that $t[t_1, ..., t_n] \in SSV$. It is only necessary to show that $t(t_1, ..., t_n) \in SV$ to implicitly define $\mu\colon SS \to S$; then the commutativity of Figures 1 and 2 are easy verifications. As $t(\mathbf{v}), t_1, ..., t_n \in SV$, there are schemes $s, s_1, ..., s_n$ such that the first components of these schemes, say $G_i(\mathbf{v}_i) = t_i$, have $t(\mathbf{v}), t_1, ..., t_n$ as solutions, respectively. Form the scheme consisting of the disjoint union of $s, s_1, ..., s_n$ together with the rule $G_{n+1}(\mathbf{v}_{n+1}) = G_0(G_1(\mathbf{v}_1), G_2(\mathbf{v}_2), ..., G_n(\mathbf{v}_n))$. Then a solution to this rule in the union scheme is $t(t_1, ..., t_n)$. ∎

4.11. COROLLARY. $\mathrm{Sch}(F)$ *is initial in the category of F-algebras in which all schemes on F have solutions.*

Consider now the class of all monads in which every scheme has a solution. To avoid trivial monads, we consider only the subclass of monads in which M is a fixed submonad of each solution monad. Technically, R is a solution monad if for each monad endomorphism $\phi: M_\Phi \to M_\Phi$ there is a solution morphism $g_\phi: M_\Phi \to R$ such that $g_\phi = \phi \cdot g_\phi$ and $i \cdot g_\phi$ is a monomorphism with $i: M \to M_\Phi$ the canonical injection. The free monad M is a fixed submonad of R if $i \cdot g_\phi = i_s \cdot g_{\phi_s}$ for all monad endomorphisms ϕ_s, $s \in S$. Let **R** be the class of all pairs $(\{g_\phi\}, R)$ where R is a solution monad with M a fixed submonad of R and $\{g_\phi\}$ is the set of all solution morphisms for R. The set **R** becomes a category by defining the **R**-morphisms $p: (\{g_\phi\}, R) \to (\{g'_\phi\}, R')$ to be monad morphisms $p: R \to R'$ such that $g_\phi \cdot p = g'_\phi$ for each ϕ.

4.12. PROPOSITION. $(\{k_s \cdot \chi_s\}, P)$ *is the initial object of* **R**.

Proof. For $(\{g_\phi\}, R)$ in **R** and $s = s_\phi$, g_ϕ factors uniquely through the coequalizer (k_s, K_s) as $g_\phi = k_s \cdot h_\phi$. Now $\bar{\phi}_1 \cdot h_{\phi_1} = \bar{\phi}_2 \cdot h_{\phi_2}$ for all $\phi_1, \phi_2 \in S$, so each h_ϕ factors uniquely through $(\{\chi_s\}, P)$ as $h_{\phi_s} = \chi_s \cdot f$ for some monad morphism $f: P \to R$ which depends only on $(\{g_\phi\}, R)$. ∎

For the purpose of stating the next theorem, let Φ_s denote the canonical insertion of the function variables into M_{Φ_s}. Say that the solutions to scheme s factor through solution monad Q if for every **R**-morphism $p: P \to R$ there is a monad morphism $q: Q \to R$ such that $\Phi_s \cdot k_s \cdot \chi_s \cdot p = \Phi_s \cdot k_s \cdot \chi_s \cdot \bar{p} \cdot q$, where $\bar{p}: P \to Q$ is the unique **R**-morphism. Note that $q: Q \to R$ is not required to be an **R**-morphism, only that the images of the function variables, i.e., the solutions, factor through Q. Figure 18 diagrams this.

4.13. THEOREM. *Every scheme has a solution in* Sch(F). *The solutions to all schemes factor through* Sch(F). *Moreover every Greibach scheme has a unique solution in* Sch(F).

Proof. The last statement is known, [9]. We prove the first two. By Proposition 4.12, there is a unique morphism $\pi: P \to S = \mathrm{Sch}(F)$ such that every scheme s factors through π as $c_s = k_s \cdot \chi_s \cdot \pi$. By definition, for every $t \in SV$ there is a scheme s such that t is a solution of s. Therefore πV is surjective for all V so that π is extremal epi. For each **R**-morphism $p: P \to R$ define $\sigma: S \to R$ by $\pi[G] \mapsto {}^{\sigma}p[G]$, where $[G]$ is the equivalence class of $G \in \Phi = \bigsqcup \Phi_s$ in P. Since π is epi σ is everywhere defined. It is easily seen to be fixed on M and one may verify that σ is a monad morphism. ∎

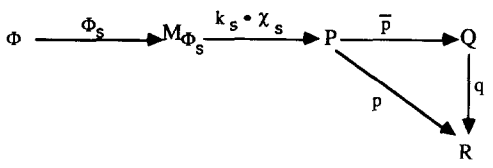


FIGURE 18

Note that $\mathrm{Sch}(F)$ and P are not isomorphic because of the lack of unique solutions to Greibach schemes in P. Namely, P will usually contain several solutions for a given scheme; this is due to the fact that P is initial with respect to *all* F-algebras in which s has solutions, and in some of these F-algebras, solutions may be nonunique since we do not assume any special property (ordering or metric) of the algebra. In Example 3.15, for instance, $\mathrm{Sch}(F) = A(F)$ is obviously embedded in $K\varnothing \subset P$. So P has more elements than $\mathrm{Sch}(F)$. Let **U** be the subcategory of **R** for which all Greibach schemes have unique solutions in all the solution monads. It is now easily seen that $(\{c_s\}, \mathrm{Sch}(F))$ is initial in **U**.

Constructions having similar goals within a universal algebra framework may be found in [30] for ordered magmas and n-rational schemes and in [49] for iterative schemes with unique solutions. Also see [15, 48].

5. Solving Schemes in Algebras with Relations

In this section we show that the algebraic techniques for free monads of the previous sections lead directly to obtaining scheme solutions in monads obtained by congruence on the free monads. Therefore scheme solutions may be obtained in any algebra presented by generators and relations.

For example, the equations

$$\begin{aligned} x + (y + z) &= (x + y) + z, \\ x + y &= y + x, \\ x + x &= x, \\ x + \delta &= x, \end{aligned} \tag{US}$$

are the relations defining unitary semilattices with unit δ. The unitary semilattices are a standard framework for the study of nondeterministic computation [38]. The recursion schemes over unitary semilattices provide models of nondeterministic computation, and the above equations (US) suffice to study *run-time choice* nondeterministic computation [12]. In the study of *call-time choice* nondeterministic computation, one additionally encounters the strict linearity equations

$$\begin{aligned} f(x + y) &= f(x) + f(y), \\ f(\delta) &= \delta, \end{aligned}$$

for each unary base function symbol; the strict bilinearity equations

$$\begin{aligned} g(x + y, z) &= g(x, z) + g(y, z), \\ g(x, y + z) &= g(x, y) + g(x, z), \\ g(x, \delta) &= \delta = g(\delta, y), \end{aligned}$$

for each binary base function symbol other than $+$ itself, and corresponding strict multilinearity equations for each n-ary base function symbol [45].

The techniques presented here nicely provide solutions for run-time choice schemes but fail to do so for call-time choice schemes in that there is no way, in the current framework, to give linearity equations for the function variables. The deeper reason is that we are considering full Kleene substitution throughout the paper, and this is the wrong computational method for call-time choice semantics, cf. [12].

Another example of particular interest is the process algebras of [14], as fully satisfactory monotonic approximation orders do not exist in process algebras. Therefore solutions to recursion schemes in process algebras cannot be obtained as least fixed points.

We list only some of the defining equations for process algebras here, enough to make a particular point. The signature F of a process algebra includes a binary addition with Eq. (US), a binary multiplication, and constants δ and τ. The defining equations of interest here are

$$\begin{aligned} x \cdot \tau &= x, \\ x + \tau \cdot x &= \tau \cdot x. \end{aligned} \tag{T}$$

The equation $\tau \cdot (x+y) + x = \tau \cdot (x+y)$ is a consequence of (US) and (T).

Consider the first-order recursion scheme

$$s\colon G = \tau(G+\tau).$$

In $\mathrm{Sch}(F)$ the canonical solution is the equivalence class $\{G, \tau \cdot (G+\tau), \tau \cdot (\tau \cdot (G+\tau)+\tau), \ldots\}$. In the monad we construct in this section, this equivalence class is reduced by (US) and (T) and so is the two-element set $E = \{G, \tau \cdot (G+\tau)\}$, since

$$\tau \cdot (G+\tau) = \tau \cdot (\tau \cdot (G+\tau)+\tau).$$

Now all solutions to scheme s in a free process algebra are of the form $\tau \cdot (y+\tau)$ for arbitrary y [13]. The construction we give in this section will in effect provide process algebras $\hat{K}X$ for each set of generators X. Each $\hat{K}X$ is on the signature $F' = F \cup \{E\}$ with E satisfying the equation $E = \tau \cdot (E+\tau)$ and the canonical solution to s in $\hat{K}X$ is E. The construction of $\hat{S}$ provides the freest possible extension of this style to the usual signature so that all recursion schemes have solutions.

A congruence on M is described by a monad Q and a pair of monad morphisms $p, q\colon Q \to M$. At the level of **Set**, think of Q as a set of ordered pairs listing the congruential elements and think of p and q as the projections into M. Now $\rho\colon M \to T$ is regular epi iff there exists such a pair $p, q\colon Q \to M$ such that $(\rho, T) \approx \mathrm{Coeq}(p, q)$.

Let M be a free monad as before, and let the monad morphism $\rho\colon M \to T$ be regular epi. Such morphisms are the canonical epimorphisms of a congruence relation in the universal algebraic sense, sending each M-structure into its

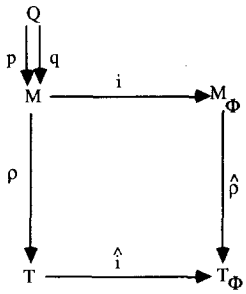


FIGURE 19

equivalence class. So $\rho_V: MV \to TV$ sends M-trees over V to equivalence classes of M-trees over V.

Given the regular epi $\rho: M \to T$, define $(\hat{\imath}: T \to T_\Phi,\ \hat{\rho}: M_\Phi \to T_\Phi, T_\Phi)$ as the pushout in Fig. 19, where $i: M \to M_\Phi$ is the canonical injection. Since ρ is regular epi, so is $\hat{\rho}$, [39, Section 21.13^{op}]. A standard proof in category theory shows that $(\hat{\rho}, T_\Phi) \approx \text{Coeq}(p \cdot i, q \cdot i)$, see Proposition A.1 in the Appendix.

We now extend Fig. 19 by defining the morphisms denoted by ---, — · — · and · · · arrows in Fig. 20.

As $\hat{\rho}$ coequalizes $p \cdot i$ and $q \cdot i$ and $i\phi = i$,

$$p \cdot i \cdot \phi \cdot \hat{\rho} = p \cdot i \cdot \hat{\rho} = q \cdot i \cdot \hat{\rho} = q \cdot i \cdot \phi \cdot \hat{\rho},$$

and so there exists a unique morphism mediating from $\text{Coeq}(p \cdot i, q \cdot i)$ to $\phi \cdot \hat{\rho}$, which we denote by $\hat{\phi}: T_\Phi \to T_\Phi$; the mediation being $\hat{\rho} \cdot \hat{\phi} = \phi \cdot \hat{\rho}$. Let $(\hat{k}, \hat{K}) \approx \text{Coeq}(\hat{\phi}, \text{id})$. Then $\hat{\rho} \cdot \hat{\phi} \cdot \hat{k} = \phi \cdot \hat{\rho} \cdot \hat{k} = \hat{\rho} \cdot \hat{k}$ implies there is a unique morphism, $\zeta: K \to \hat{K}$, mediating between the coequalizers. By [39, Section 29B^{op}], ζ is extremal epi. Moreover, scheme $s = s_\phi$ has a canonical solution in $\hat{K}$.

5.1. PROPOSITION. $(\zeta, \hat{K}) \approx \text{Coeq}(p \cdot i \cdot k,\ q \cdot i \cdot k)$; *hence* ζ *is a regular epi.*

Proof. Consider Fig. 21. Let $(k', K') \approx \text{Coeq}(p \cdot i \cdot k, q \cdot i \cdot k)$. The equality $pi \cdot kk' = qi \cdot kk'$ implies that there exists a unique $\beta: T_\Phi \to K'$ such that $kk' = \hat{\rho}\beta$, since $(\hat{\rho}, T_\Phi) \approx \text{Coeq}(pi, qi)$. Then $\hat{\rho}\hat{\phi}\beta = \phi\hat{\rho}\beta = \phi kk' = kk' = \hat{\rho}\beta$. Since $\hat{\rho}$ is epi,

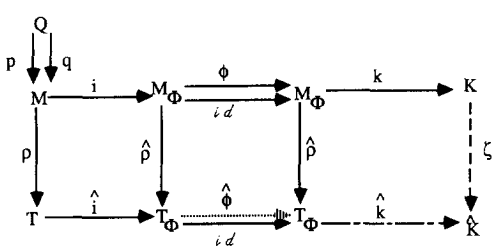


FIGURE 20

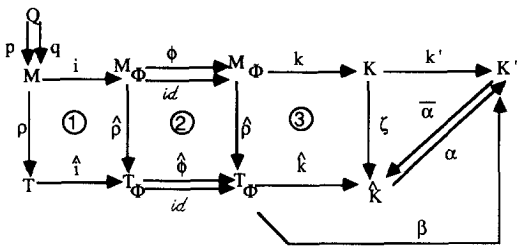


FIGURE 21

$\hat{\phi}\beta = \beta$, hence by the coequalizer property of $\hat{K}$ there exists a unique $\alpha: \hat{K} \to K'$ such that $\hat{k}\alpha = \beta$.

Now $p \cdot i \cdot k \cdot \zeta = q \cdot i \cdot k \cdot \zeta$ by the commutativity of squares labeled 1, 2, 3 in Fig. 21. By the coequalizer property of K' there exists a unique $\bar{\alpha}: K' \to \hat{K}$ such that $k'\bar{\alpha} = \zeta$. Now $kk'\bar{\alpha}\alpha = k\zeta\alpha = \hat{\rho}\hat{k}\alpha = \hat{\rho}\beta = kk'$, so as kk' is epi, $\bar{\alpha}\alpha = \mathrm{id}$. Similarly, $\hat{\rho}\hat{k}\alpha\bar{\alpha} = \hat{\rho}\beta\bar{\alpha} = kk'\bar{\alpha} = k\zeta = \hat{\rho}\hat{k}$ and $\hat{\rho}\hat{k}$ being epi, $\alpha\bar{\alpha} = \mathrm{id}$. Hence $(\zeta, \hat{K}) \approx \mathrm{Coeq}(pik, qik)$. ∎

5.2. COROLLARY. *$(\zeta, \hat{k}, \hat{K})$ is the pushout of $(M_\Phi, k, \hat{\rho})$.*

Proof. Consider any K' and morphisms $\gamma: K \to K'$, $\beta: T_\Phi \to K'$. Let (K', γ, β) be such that $k \cdot \gamma = \hat{\rho} \cdot \beta$, see Fig. 22.

Then $p \cdot i \cdot k \cdot \gamma = p \cdot i \cdot \hat{\rho} \cdot \beta = q \cdot i \cdot \hat{\rho} \cdot \beta = q \cdot i \cdot k \cdot \gamma$. Hence, by the coequalizer property of $(\zeta, \hat{K})$, there exists a unique $m: \hat{K} \to K'$ such that: $\zeta \cdot m = \gamma$. Now we have $\hat{\rho} \cdot \hat{k} \cdot m = k \cdot \zeta \cdot m = k \cdot \gamma = \hat{\rho} \cdot \beta$. As $\hat{\rho}$ is epi, $\hat{k} \cdot m = \beta$. Hence $(\zeta, \hat{k}, \hat{K})$ has the universal property of the pushout. ∎

In order to carry out the pushout construction of Section 4 to the factor structures T_Φ, we need the following.

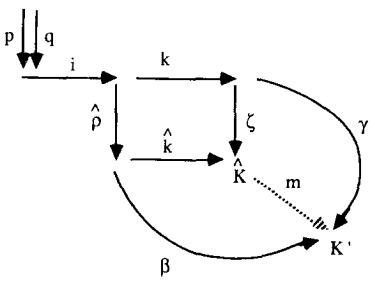


FIGURE 22

5.3. PROPOSITION. *In Fig. 23* $\hat{\imath}: T \to T_\Phi$ *is a monomorphism.*

Proof (Due to M. Main). Let T_Ω be a monad such that $\tau: T \to T_\Omega$ is an injection and the function symbols Ω_n for each rank n form congruence classes according to the equations

$$\Omega_n(t_1, ..., t_n) \equiv \Omega_m(x_1, ..., x_m),$$

where $t_1, ..., t_n$ are congruence classes of arity m in $T_\Omega X$ over the constants in F_0 and variables $x_1, ..., x_m$ in X. Define $\hat{\tau}: M_\Phi \to T_\Omega$ by

$$\hat{\tau}(t) = \begin{cases} \tau(\rho(t)) & \text{if } t \text{ is in the image of } i, \\ \Omega_n & \text{otherwise, where } t \text{ has arity } n. \end{cases}$$

That $i \cdot \hat{\tau} = \rho \cdot \tau$ is immediate; hence there is a unique mediating morphism $m: T_\Phi \to T_\Omega$ such that $\hat{\rho} \cdot m = \hat{\tau}$ and $\hat{\imath} \cdot m = \tau$. As τ is a monomorphism by construction, it follows that $\hat{\imath}$ is a monomorphism. ∎

We now proceed to do the pushout construction analogous to the construction of P which will provide us with the monad $\hat{P}$ in which every scheme has a solution modulo ρ. A final pushout construction starting with $\hat{P}$ and $S = \mathrm{Sch}(F)$ will then provide $\hat{S}$ as the modulo ρ correspondent to $\mathrm{Sch}(F)$. We need a few preliminary results.

5.4. THEOREM. *Let* $\Phi = \Phi_1 \sqcup \Phi_2$. T_Φ *is the pushout of* $(T, \hat{\imath}_1, \hat{\imath}_2)$.

Proof. Consider Fig. 24, where α, $\bar{\alpha}$, α_1, α_2 and R are defined in the subsequent text.

By the definition of T_Φ, T_{Φ_j}, the diagrams in Figs. 25 and 26 are pushouts.

For $j = 1, 2$, let i, i_j, i'_j be the canonical injections. By the universal property of M, M_{Φ_j}, $i = i_j \cdot i'_j$. As T_Φ is the pushout of i and ρ, $i \cdot \hat{\rho} = i_j \cdot i'_j \cdot \hat{\rho} = \rho \cdot \hat{\imath}$. By the pushout property of T_{Φ_j}, there exists unique $\hat{\imath}'_j$ such that $\hat{\imath}_j \cdot \hat{\imath}'_j = \hat{\imath}$ and $i'_j \cdot \hat{\rho} = \hat{\rho}_j \cdot \hat{\imath}'_j$. Thus everything commutes in Fig. 27; now the left square and the big rectangle are pushouts, hence by [39, Section 21Eb^{op}] the right square a pushout.

Let us prove now that T_Φ is the pushout of $(T, \hat{\imath}_1, \hat{\imath}_2)$. Let α_1, α_2 be such that $\hat{\imath}_1 \cdot \alpha_1 = \hat{\imath}_2 \cdot \alpha_2$. Hence $\rho \cdot \hat{\imath}_1 \cdot \alpha_1 = \rho \cdot \hat{\imath}_2 \cdot \alpha_2 = i_2 \cdot \hat{\rho}_2 \cdot \alpha_2 = i_1 \cdot \hat{\rho}_1 \cdot \alpha_1$. M_Φ being the pushout of (M, i_1, i_2), there exists a unique α such that

$$i'_j \cdot \alpha = \hat{\rho}_j \cdot \alpha_j \qquad \text{for } j = 1, 2. \tag{5.4.1}$$

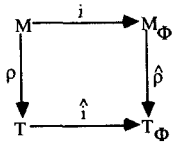


FIGURE 23

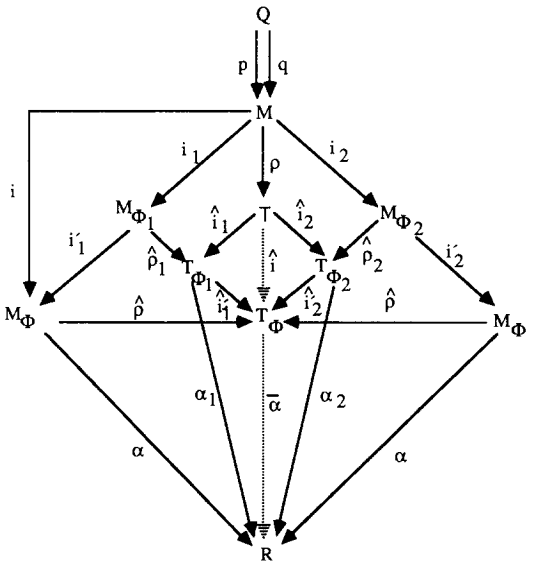
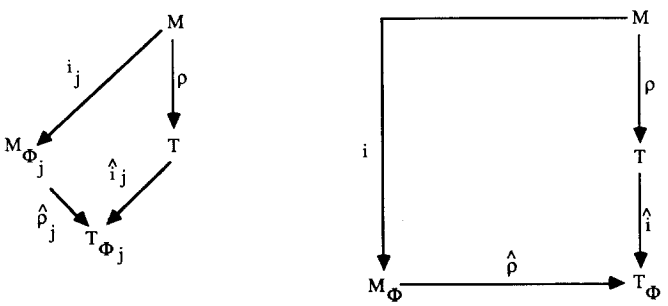


FIGURE 24



FIGURES 25 AND 26

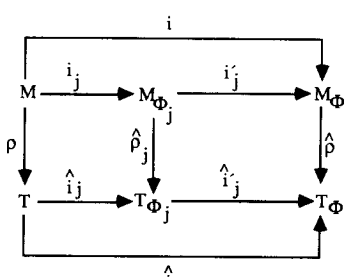


FIGURE 27

Since T_Φ is the pushout of $(M_{\Phi_1}, i'_1, \hat{\rho}_1)$, there exists a unique $\bar{\alpha}$ such that

$$\hat{\imath}'_1 \cdot \bar{\alpha} = \alpha_1 \qquad \text{and} \qquad \hat{\rho} \cdot \bar{\alpha} = \alpha. \tag{5.4.2}$$

Now, $\hat{\rho}_2 \cdot \hat{\imath}'_2 \cdot \bar{\alpha} = i'_2 \cdot \hat{\rho} \cdot \bar{\alpha}$ (definition of $\hat{\imath}'_2$) $= i'_2 \cdot \alpha = \hat{\rho}_2 \cdot \alpha_2$. As is $\hat{\rho}_2$ is epi, we also have $i'_2 \cdot \bar{\alpha} = \alpha_2$, whence $\hat{\imath}'_j \cdot \bar{\alpha} = \alpha_j$, $j = 1, 2$.

To show the uniqueness of $\bar{\alpha}$, let β be such that $\hat{\imath}'_j \beta = \alpha_j$, $j = 1, 2$. Then $i'_j \cdot \hat{\rho} \cdot \beta = \hat{\rho}_j \cdot \hat{\imath}'_j \cdot \beta = \hat{\rho}_j \cdot \alpha_j$. By (5.4.1), α is the unique arrow satisfying this equality hence $\hat{\rho}\beta = \alpha$. Now β satisfies $\hat{\imath}'_1 \cdot \beta = \alpha_1$ and $\hat{\rho}\beta = \alpha$ hance by (5.4.2) $\beta = \bar{\alpha}$. ∎

The next step in the modulo ρ construction is to generalize from two schemes to an arbitrary collection. In what follows, let J be the index set and we take j and l to be indices from J without further mention. We have $(k_j, K_j) \approx \mathrm{Coeq}(\phi_j, \mathrm{id})$ and $(\hat{k}_j, \hat{K}_j) \approx \mathrm{Coeq}(\hat{\phi}_j, \mathrm{id})$ is the modulo ρ correspondent. The system $(\{\chi_j\}, P)$ is the pushout of $(M, \{\psi_j\})$ and the system $(\{\hat{\chi}_j\}, \hat{P})$ is the pushout of $(T, \{\hat{\psi}_j\})$, where $\psi_j = i \cdot k_j$ and $\hat{\psi}_j = \hat{\imath} \cdot \hat{k}_j$.

5.5. PROPOSITION. *The pushout of* (M, ρ, ψ_j) is $(\hat{\psi}_j, \zeta_j, \hat{K}_j)$ *for each* j.

Proof. Compose the pushout squares of 5.2 and 5.4, [39, Section 21Ea^{op}]. ∎

Consider Fig. 28 which illustrates the two multiple pushouts we are considering.

5.6. PROPOSITION. *There exists a unique* $\bar{\rho}: P \to \hat{P}$ *such that* $\zeta_j \cdot \hat{\chi}_j = \chi_j \cdot \bar{\rho}$ *for each* j.

Proof. For every j, l,

$$\psi_j \cdot \zeta_j \cdot \hat{\chi}_j = \rho \cdot \hat{\psi}_j \cdot \hat{\chi}_j = \rho \cdot \hat{\psi}_l \cdot \hat{\chi}_l = \psi_l \cdot \zeta_l \cdot \hat{\chi}_l$$

so, as P is a pushout, the unique mediating morphism $\bar{\rho}$ exists. ∎

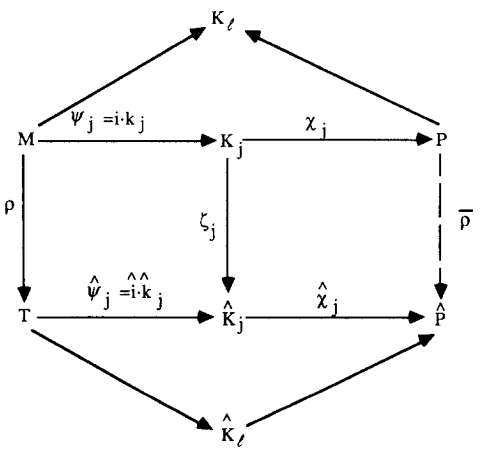


FIGURE 28

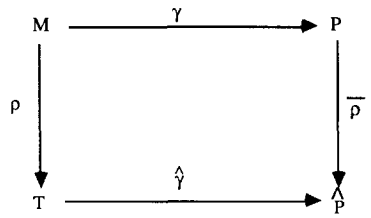


FIGURE 29

5.7. DEFINITION. As $\psi_j \cdot \chi_j = \psi_l \cdot \chi_l$ for all j, l, let $\gamma: M \to P$ denote this composite morphism, the "diagonal" of the pushout. Similarly, let $\hat{\gamma}: T \to \hat{P}$ denote the composite morphism $\hat{\psi}_j \cdot \hat{\chi}_j$.

We next show that Fig. 29 is a pushout square.

5.8. PROPOSITION. *The pushout of* (M, ρ, γ) *is* $(\hat{\gamma}, \bar{\rho}, \hat{P})$.

Proof. Let (α, β, Q) be the pushout of (M, ρ, γ). As $\gamma \cdot \alpha = \psi_j \cdot \chi_j \cdot \alpha$ for any j, and $(\hat{\psi}_j, \zeta_j, \hat{K}_j)$ is the pushout of (M, ρ, ψ_j), and $\gamma \cdot \alpha = \rho \cdot \beta$, there is a unique $m_j: \hat{K} \to Q$ for which

$$\gamma \cdot \alpha = \psi_j \cdot \chi_j \cdot \alpha = \psi_j \cdot \zeta_j \cdot m_j = \rho \cdot \beta = \rho \cdot \hat{\psi}_j \cdot m_j.$$

As ρ is epi, $\beta = \hat{\psi}_j \cdot m_j$. As $\beta = \hat{\psi}_l \cdot m_l$ for all l, β factors uniquely through the pushout $\hat{P}$ via some mediating morphism $\delta: \hat{P} \to Q$ as $\beta = \hat{\psi}_j \cdot \hat{\chi}_j \cdot \delta = \hat{\gamma} \cdot \delta$. By the hypothesis that (α, β, Q) is the pushout of (M, ρ, γ) there is a unique $\bar{\delta}: Q \to \hat{P}$ such that $\hat{\gamma} = \beta \cdot \bar{\delta}$ and $\bar{\rho} = \alpha \cdot \bar{\delta}$. Since $\gamma \cdot \alpha = \rho \cdot \hat{\gamma} \cdot \delta = \gamma \cdot \bar{\rho} \cdot \delta = \rho \cdot \beta$ and as both $\bar{\rho}$ and δ are unique mediating morphisms, $\alpha = \bar{\rho} \cdot \delta$. Then $\bar{\rho} = \bar{\rho} \cdot \delta\bar{\delta}$ and as $\bar{\rho}$ is epi, $\delta\bar{\delta} = \mathrm{id}$. By similar reasoning, $\alpha = \alpha \cdot \delta\bar{\delta}$. As α is the pushout of the epi ρ, α is epi, and $\delta\bar{\delta} = \mathrm{id}$. ∎

As in the free case in Corollary 4.6, any scheme on $\hat{P}$ has a canonical solution in $\hat{P}$, i.e., $\hat{P}$ is algebraically closed.

The final step of the modulo ρ construction is a pushout from P towards $\mathrm{Sch}(F)$ along $\hat{P}$, the universal solution monad of the modulo ρ schemes. Recall that $S = \mathrm{Sch}(F)$ and $\pi: P \to S$ is the unique mediating morphism of Theorem 4.13. The diagram of interest is shown in Fig. 30.

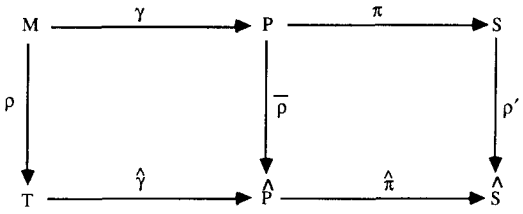


FIGURE 30

5.9. DEFINITION. *Let* $(\hat{\pi}, \rho', \hat{S})$ be the pushout of $(P, \pi, \bar{\rho})$.

5.10. PROPOSITION. *The pushout of* $(M, \rho, \gamma \cdot \pi)$ *is* $(\hat{\gamma} \cdot \hat{\pi}, \rho', \hat{S})$.

Proof. Compose the pushout squares of 5.8 and 5.9 [39, Section 21Ea^{op}]. ∎

5.11. *Remark.* The monad $\hat{S}$ is *not* isomorphic to what is usually denoted by $\mathrm{Sch}(F)/\rho$ in program scheme theory; some elements which are collapsed in the latter are not collapsed in the former. This is due to the fact that the equivalence on $\mathrm{Sch}(F)/\rho$ is defined by extending ρ by continuity, using the ordering, from finite trees to infinite trees. Since we introduced neither an ordering nor a metric here, no continuity extension can be defined, hence no extra collapsing will occur. More precisely, let ρ^a be defined on M^∞ as in [34], let $i^\infty: \mathrm{Sch}(F) \to M^\infty$ be the canonical injection and let $\rho^\infty: M^\infty \to M^\infty/\rho^a$ be the canonical surjection. Define $\mathrm{Sch}(F)/\rho = \rho^\infty(i^\infty(\mathrm{Sch}(F))$.

Take now $F = \{f, g, \Omega\}$,

$$\rho: f \cdot g \equiv g \cdot f \qquad \text{and} \qquad f(\Omega) \equiv g(\Omega);$$

then if

$$G = g(G)$$
$$H = f(H)$$

is a scheme, $G \rho^a H$, hence G and H are collapsed in $\mathrm{Sch}(F)/\rho$ but G and H are not collapsed in $\hat{S}$.

5.12. PROPOSITION. $\bar{\rho}$ *and* ρ' *are regular epi.*

Proof. The pushout of regular epis is a regular epi; then, by diagram chasing in Fig. 30, $\bar{\rho}$ and then ρ' are at the end of a sequence of pushouts of regular epis. ∎

5.13. COROLLARY. $\hat{\pi}: \hat{P} \to \hat{S}$ *is extremal epi.*

Proof. $\pi \cdot \rho' = \bar{\rho} \cdot \hat{\pi}$ is extremal epi as ρ' is regular epi, therefore extremal epi, and π is extremal epi, therefore epi. Therefore $\hat{\pi}$ is extremal epi. ∎

Define **U** to be the subcategory of the category of solution monads **R** consisting of the monads in which solutions satisfy the original congruence $(p: Q \to M, q: Q \to M)$. Specifically, $(\{g_\phi\}, \hat{R})$ is an object of **U** if there is a monad morphism $r: M \to \hat{R}$ such that $p \cdot r = q \cdot r$ and $i \cdot g_\phi = r$ for each g_ϕ. The morphisms of **U** are all the morphisms of **R** between objects of **U**.

5.14. PROPOSITION $(\{k_s \cdot \chi_s \cdot \bar{\rho}\}, \hat{P})$ *is initial in* **U**.

Proof. For $(\{g_s\}, \hat{R})$ in **R**, $g_s = k_s \cdot \chi_s \cdot f$ for $f: P \to \hat{R}$ by Proposition 4.12. The

morphism $r: M \to \hat{R}$ factors uniquely through ρ as $r = \rho \cdot \bar{r}$. As $i \cdot g_s = i \cdot k_s \cdot \chi_s \cdot f = \rho \cdot \bar{r}$ and $(\bar{\rho}, \hat{\imath} \cdot \hat{k}_s \cdot \hat{\chi}_s, \hat{P})$ is the pushout of $(M, \rho, i \cdot k_s \cdot \chi_s)$ there is a unique morphism $\hat{f}: \hat{P} \to \hat{R}$ such that $f = \bar{\rho} \cdot \hat{f}$ and $\bar{r} = \hat{\imath} \cdot \hat{k}_s \cdot \hat{\chi}_s \cdot \hat{f}$. ∎

5.15. THEOREM. *The solutions to all schemes factor through* $\hat{S}$.

Proof. By Proposition 5.14 and Corollary 5.13, the factorization technique used in Theorem 4.13 applies in this case as well. ∎

From this theorem and Remark 5.11, we see that $\mathrm{Sch}(F)/\rho$ is not a monad. The intuitive reason is that identifications are made in $\mathrm{Sch}(F)/\rho$: in a sense, too many elements are collapsed thus destroying freeness.

6. CONCLUSION

We have described a purely algebraic and general solution for the semantics of recursive schemes. This solution does not involve extra structures such as order or topology. The use of category theoretic tools to express it makes it most general and most straightforward. In essence, we first constructed solutions to schemes using coequalizers, then pasted the solutions together using pushouts. This construction was then factored through a relation ρ to model the construction of solutions in the case when relations hold between base function symbols. Our construction straightforwardly generalizes to yield solutions for nondeterministic schemes; it also does not have the drawbacks of usual order-theoretic semantics and should thus more easily generalize for giving a semantics for communicating processes. For generalizations in other directions, consider [40, Chap. VI].

APPENDIX: SOME NOTATIONS AND CATEGORY THEORY

In Sections 4 and 5, we have some diagram chasing proofs. These are made easier to follow by introducing composition in lexical order using the $\cdot$ notation of [46]. For morphisms $f_1: A_1 \to A_2$ and $f_2: A_2 \to A_3$, the composition is denoted $f_1 \cdot f_2: A_1 \to A_3$ and we frequently elide the dot to write $f_1 f_2$ instead of $f_1 \cdot f_2$. The composition in the usual mathematical order is always denoted by $\circ$, as in $f_2 \circ f_1$.

There are slight inconsistencies in terminology in the standard texts. We choose to use the definitions from [39]. Since this text does not use the dot notation, we give the required definitions using dot notation for ease of reference.

A morphism f is *mono* if for all morphisms a, b for which $a \cdot f = b \cdot f$ we have $a = b$. If $(m \cdot p)$ is mono then m is mono. The morphism $m: E \to A$ is an *equalizer* of the parallel morphisms $a, b: A \to B$ if it is a pre-equalizer, i.e., $m \cdot a = m \cdot b$, and for all pre-equalizers $f: C \to A$ there is a unique morphism $\hat{f}: C \to E$ such that $f = \hat{f} \cdot m$. Every equalizer is mono. We write $(E, m) \approx \mathrm{Equ}(k, l)$. The equalizer property is pictured in Fig. 31a.

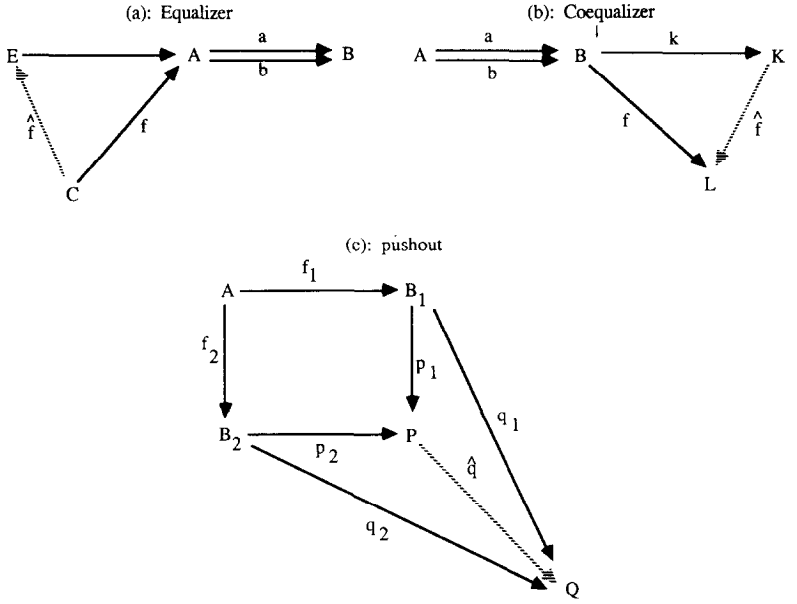
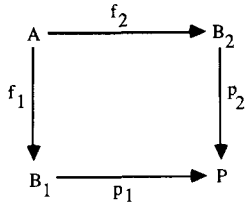


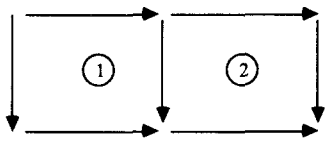
FIGURE 31

A morphism f is *epi* if for all morphisms k, l for which $f \cdot k = f \cdot l$ we have $k = l$. If $(p \cdot e)$ is epi then e is epi. The morphism $k: B \to K$ is a *coequalizer* of the parallel morphisms $a, b: A \to B$ if it is a pre-coequalizer, i.e, $a \cdot k = b \cdot k$, and for all pre-coequalizers $f: B \to L$ there is a unique morphism $\hat{f}: K \to L$ such that $f = k \cdot \hat{f}$. Every coequalizer is epi. The coequalizer property is pictured in Fig. 31b. We write $(k, K) \approx \text{Coeq}(a, b)$. A morphism $k: B \to K$ is *regular epi* if there exists some pair of parallel morphisms a, b such that $(k, K) \approx \text{Coeq}(a, b)$. A morphism e is *extremal epi* if it is an epimorphism and if $e = f \cdot m$ where m is mono, then m must be an isomorphism. Every regular epi is extremal epi.

The pair of morphisms $p_1: B_1 \to P$, $p_2: B_2 \to P$ is a *pushout* of the morphisms $f_1: A \to B_1, f_2: A \to B_2$ if $f_1 \cdot p_1 = f_2 \cdot p_2$ and for every pair of morphisms $q_1: B_1 \to Q$, $q_2: B_2 \to Q$ such that $f_1 \cdot q_1 = f_2 \cdot q_2$ there is a unique morphism $\hat{q}: P \to Q$ such that $q_1 = p_1 \cdot \hat{q}$ and $q_2 = p_2 \cdot \hat{q}$. We say that (p_1, p_2, P) is a pushout and write $(p_1, p_2, P) \approx \text{Pushout}(A, f_1, f_2)$. The pushout property is pictured in Fig. 31c. The pushout of an epi is epi and the pushout of a regular epi is regular epi: Specifically, if f_1 is (regular) epi, so is p_2. In **Set** the pushout of a mono is mono. The configuration in Fig. 32 is called a pushout square. In Fig. 33 if the squares 1 and 2 are pushout squares so is the outer rectangle, i.e., pushouts compose by "pasting edges." If the outer rectangle and square 1 are pushout squares, so is square 2.

A *sink* is a pair $(\{f_j\}_J, Z)$ with $\{f_j: Z_j \to Z\}$ a J-indexed family of morphisms with codomain Z. A sink is an epi-sink provided the f_j can be simultaneously cancelled from the left, i.e., provided that for any pair of parallel morphisms $k, l: Z \to X$





FIGURES 32 AND 33

such that $f_j \cdot k = f_j \cdot l$ for all $j \in J$, it follows that $k = l$. Every colimit is an epi-sink. Specifically: If $(k, K) \approx \text{Coeq}(a, b)$ then $(\{k\}, K)$ is an epi-sink. If (p_1, p_2, P) is a pushout then $(\{p_1, p_2\}, P)$ is an epi-sink.

The morphism $k: B \to K$ is a *multiple coequalizer* of the family of parallel morphisms $\{a_i: A \to B\}_J$ if it is a pre-(multiple coequalizer), i.e., $a_i \cdot k = a_j \cdot k$ for all $i, j \in J$, and for all pre-(multiple coequalizers) $f: B \to L$ there is a unique morphism $\hat{f}: B \to L$ such that $f = k \cdot \hat{f}$. Every multiple coequalizer is epi. The family of morphisms $\{p_j: B_j \to P\}_J$ is a *multiple pushout* of the family $\{f_j: A \to B_j\}_J$ if $f_i \cdot p_i = f_j \cdot p_j$ for every $i, j \in J$ and for every family of morphisms $\{q_j: B_j \to Q\}_J$ such that $f_i \cdot q_i = f_j \cdot q_j$ for every $i, j \in J$ there is a unique morphism $\hat{q}$ such that $q_j = p_j \cdot \hat{q}$ for all $j \in J$.

The following proposition illustrates some of these concepts.

A.1. PROPOSITION. *Let $m: A \to B_1$ be a morphism and let $e: A \to B_2$ be a regular epi. Specifically, let $(e, B_2) \approx \text{Coeq}(p, q)$. Let $(\hat{e}, \hat{m}, P)$ be the pushout of (A, m, e). Then $(\hat{e}, P) \approx \text{Coeq}(pm, qm)$.*

Proof. The morphism $\hat{e}$ is regular epi from [39, Section 21.13^{op}] but we give the explicit proof without relying on that fact. Let $(c, C) \approx \text{Coeq}(pm, qm)$ to show $(\hat{e}, P) \approx (c, C)$. We have Fig. 34 with the morphisms a, b, d yet to construct. As $pm\hat{e} = pe\hat{m} = qe\hat{m} = qm\hat{e}$, $(\hat{e}, P)$ is a pre-coequalizer of pm and qm so there is a unique $b: C \to P$ with $cb = \hat{e}$. Further, (mc, C) is a pre-coequalizer of p and q so there is a unique $a: B_2 \to C$ such that $ea = mc$. As $(\hat{e}, \hat{m}, P)$ is the pushout, there is a unique

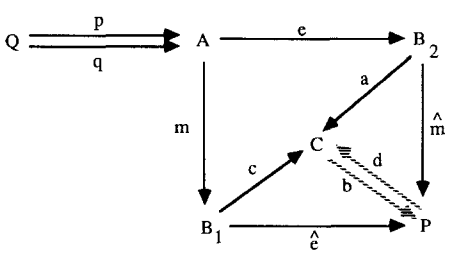


FIGURE 34

morphism $d: P \to C$ such that $c = \hat{e}d$ and $a = \hat{m}d$. Now $e\hat{m} = m\hat{e} = mcb = eab$ and as e is epi, $\hat{m} = ab$. The pushout $(\hat{e}, \hat{m}, P)$ is an epi-sink so the equations

$$\hat{m} = ab = \hat{m} \cdot db,$$

$$\hat{e} = cb = \hat{e} \cdot db$$

jointly imply that $db = \mathrm{id}$. The calculation $c = \hat{e}d = cbd$ implies $bd = \mathrm{id}$ as c is epi. Therefore $b: C \to P$ and $d: P \to C$ are isomorphisms. ∎

While these definitions make diagram-chasing proofs general, easy to write, and fairly easy to follow due to the suppression of unnecessary detail, the abstraction means that intuition must be provided outside the proof. The equalizer of $k, l: A \to B$ provides a subset of the set of points on which the functions k and l agree. If A and B are algebras or monads, the equalizing object E must be of the same type and the set of all points at which the functions k and l agree may not be of that type, so E is, in general, smaller than A. The coequalizer of $a, b: A \to B$ is obtained by finding the least congruence, $\equiv$, containing the relation $\{(a(x), b(x)) \mid x \in A\} \subseteq B \times B$. The pushout of $f_1: A \to B_1$, $f_2: A \to B_2$ has much the same flavor. Suppose f_1 and f_2 are embeddings of the subalgebra A into the algebras B_1 and B_2, respectively. Then (p_1, p_2, P) provides an "enveloping" algebra P which includes a single copy of A via $f_1 \cdot p_1 = f_2 \cdot p_2$. In general, the "copy of A" is rather mashed by the actions of f_1 and f_2. Furthermore, P has a homomorphic image of B_1 and a homomorphic image of B_2 which are as disjoint as possible given the constraints of a single homomorphic image of the subalgebra A. Finally, P is the least enveloping algebra satisfying all these constraints.

Acknowledgments

Our thanks to J. Tiuryn for an insightful afternoon and to J. Meseguer for mentioning multiple coequalizers.

References

1. S. Abramsky, On semantic foundations for applicative multiprogramming, Lecture Notes in Computer Science Vol. 154, pp. 1–14, Springer-Verlag, New York/Berlin, 1983.
2. J. Adamek, E. Nelson, and J. Reiterman, Tree constructions of free continuous algebras, *J. Comput. System Sci.* **24** (1982), 114–146.
3. ADJ*, Some fundamentals of order-algebraic semantics, Lecture Notes in Computer Science Vol. 45, pp. 153–168, Springer-Verlag, New York/Berlin, 1976.
4. ADJ, Rational algebraic theories in fixed-point solutions, *in* "Proceedings 17th IEEE Symp. Found. of Computing, Houston, 1976."
5. ADJ, Initial algebra semantics and continuous algebras, *J. Assoc. Comput. Mach.* **24** (1977), 68–95.
6. S. Alagic, Natural state transformations, *J. Comput. System Sci.* **10** (1975), 266–307.
7. M. A. Arbib and E. G. Manes, "Arrows, Structures and Functors: The Categorical Imperative," Academic Press, New York, 1975.

* ADJ denotes the set of authors {J. A. Goguen, J. W. Thatcher, E. G. Wagner, J. B. Wright}.

8. M. A. ARBIB AND E. G. MANES, The pattern-of-calls expansion is the canonical fixpoint for recursive definitions, *J. Assoc. Comput. Mach.* **29** (1982), 577–602.
9. A. ARNOLD AND M. NIVAT, Metric interpretations of infinite trees and semantics of nondeterministic recursive programs, *Theoret. Comput. Sci.* **11** (1980), 181–205.
10. A. ARNOLD AND M. NIVAT, The metric space of infinite trees: Algebraic and topological properties, *Fund. Inform.* **3** (1980), 445–476.
11. B. BANACHEWSKI AND E. NELSON, Completions of partially ordered sets, *SIAM J. Comput.* **11** (1982), 521–528.
12. D. B. BENSON, Parameter passing in nondeterministic recursive programs, *J. Comput. System Sci.* **19** (1979), 50–62.
13. D. B. BENSON AND J. TIURYN, Fixed points in free process algebras, part I, *Theoret. Comput. Sci.*, in press.
14. J. A. BERGSTRA AND J. W. KLOP, Algebra of communicating processes with abstraction, *Theoret. Comput. Sci.* **37** (1985), 77–121.
15. S. L. BLOOM, Varieties of ordered algebras, *J. Comput. System Sci.* **13** (1976), 200–212.
16. S. BLOOM, All solutions of a system of recursion equations in infinite trees and other contraction theories, *J. Comput. System Sci.* **27** (1983), 225–255.
17. S. L. BLOOM AND C. C. ELGOT, The existence and construction of free iterative theories, *J. Comput. System Sci.* **12** (1976), 305–318.
18. G. BOUDOL, Une sémantique pour les arbres non déterministes, *in* "Proceedings. 6th CAAP," Lecture Notes in Computer Science Vol. 112, pp. 147–161, Springer-Verlag, New York/Berlin, 1981.
19. P. M. COHN, "Universal Algebra," Harper & Row, New York, 1965.
20. B. COURCELLE, On recursive equations having a unique solution, *in* "Proceedings, 19th IEEE Symp, FoCS, Ann Arbor, 1978" pp. 201–213.
21. B. COURCELLE, Fundamental properties of infinite trees, *Theoret. Comput. Sci.* **25** (1983), 95–169.
22. B. COURCELLE AND J. C. RAOULT, Completions of ordered magmas, *Fund. Inform.* **3** (1980), 105–116.
23. B. COURCELLE AND M. NIVAT, Algebraic families of interpretations, *in* "Proceedings, 17th IEEE FoCS, Houston, 1976," pp. 137–146.
24. G. COUSINEAU AND M. NIVAT, On rational expressions representing infinite rational trees: application to the structure of flow charts, Lecture Notes in Computer Science Vol. 74, pp. 567–580, Springer-Verlag, New York/Berlin, 1979.
25. W. DAMM, The IO- and OI-hierarchies, *Theoret. Comput. Sci.* **20** (1982), 92–207.
26. C. C. ELGOT, Monadic computation and iterative algebraic theories, *in* "Proceedings, Logic Colloquium '73," pp. 175–230, North-Holland, Amsterdam, 1975.
27. J. ENGELFRIET, Some open questions and recent results on tree transducers and tree languages. *in* "Formal Language: Perspectives and Open Problems," pp. 241–286, Academic Press, London, 1980.
28. J. ENGELFRIET AND E. M. SCHMIDT, IO and OI, *J. Comput. System Sci.* **16** (1978), 67–99.
29. J. H. GALLIER, DPDA's in normal form and applications to equivalence problems, *Theoret. Comput. Sci.* **14** (1981), 155–186; *Theoret. Comput. Sci.* **19** (1982), 229.
30. J. H. GALLIER, *n*-rational algebras. Part 1: Basic properties and free algebras. Part II: Varieties and logic of inequations, *SIAM J. Comput.* **13** (1984), 750–794.
31. J. H. GALLIER, Recursion closed algebraic theories, *J. Comput. System Sci.* **23** (1981), 69–105.
32. S. GINALI, Regular trees and the free iterative theory, *J. Comput. System Sci.* **18** (1979), 228–242.
33. I. GUESSARIAN, On continuous completions, *in* "Proceedings 4th GI Conf.," Lecture Notes in Computer Science Vol. 67, pp. 142–152, Springer-Verlag, Berlin, 1979.
34. I. GUESSARIAN, "Algebraic Semantics," Lecture Notes in Computer Science Vol. 99, Springer-Verlag, New York/Berlin, 1981.
35. I. GUESSARIAN AND F. PARISI-PRESICCE, Iterative vs. regular factor algebras, *Sigact News* **12** (1983), 32–44.
36. M. HENNESSY AND R. MILNER, Algebraic laws for nondeterminism and concurrency, *J. Assoc. Comput. Mach.* **32** (1985), 137–162.
37. M. HENNESSY, Acceptance trees, *J. Assoc. Comput. Mach.* **32** (1985), 896–928.

38. M. HENNESSY AND G. PLOTKIN, Full abstraction for a simple parallel programming language, Lecture Notes in Computer Sciences Vol. 74, pp. 108–120, Springer-Verlag, New York/Berlin, 1979.
39. H. HERRLICH AND G. E. STRECKER, "Category Theory," 2nd. ed., Sigma Series in Pure Mathematics Vol. 1, Heldermann-Verlag, Berlin, 1979.
40. P. JOHNSTONE, "Stone Spaces," Cambridge University Press, London, 1982.
41. D. J. LEHMANN, Categories for fixed point semantics, *in* "Proceedings 17th IEEE Symp. on Foundations of Computer Science, Houston, 1976.
42. D. J. LEHMANN, On the algebra of order, *J. Comput. System Sci.* **21** (1980), 1–23.
43. D. J. LEHMANN AND M. B. SMYTH, Algebraic specification of data types, *Math. Systems Theory* **14** (1981), 97–139.
44. S. MACLANE, "Categories for the Working Mathematician," Springer-Verlag, New York/Berlin, 1971.
45. M. MAIN AND D. B. BENSON, Functional behavior of nondeterministic and concurrent programs, *Inform. and Control* **62** (1984), 144–189.
46. E. G. MANES, "Algebraic Theories," Springer-Verlag, New York/Berlin, 1976.
47. E. G. MANES AND M. A. ARBIB, "Algebraic Approaches to Program Semantics," Springer-Verlag, New York/Berlin, 1986.
48. J. MESEGUER, Varieties of chain complete algebras, *J. Pure Appl. Algebra* **19** (1980), 347–383.
49. E. NELSON, Iterative algebras, *Theoret. Comput. Sci.* **25** (1983), 67–94.
50. M. NIVAT, Langages algébriques sur le magma libre et sémantique des schémas de programmes, *in* "Automata, Languages and Programming," (M. Nivat, Ed.), pp. 293–308, North-Holland, Amsterdam, 1972.
51. G. PLOTKIN, A powerdomain construction, *SIAM J. Comput.* **5** (1976), 452–487.
52. B. K. ROSEN, Tree manipulating systems and Church–Rosser theorems, *J. Assoc. Comput. Mach.* **20** (1973), 160–187.
53. W. C. ROUNDS, On the relationships between Scott domains, synchronization trees, and metric spaces, *Inform. and Control* **66** (1985), 6–28.
54. D. SCOTT AND C. STRACHEY, Toward a mathematical semantics for computer languages, Tech. Rpt. PRG-G, Oxford Univ. Comp. Lab., 1971.
55. M. B. SMYTH, Powerdomains, *J. Comput. System. Sci.* **16** (1978), 23–36.
56. H. SCHUBERT, "Categories," Springer-Verlag, New York/Berlin, 1972.
57. J. TIURYN, Fixed points and algebras with infinitely long expressions. Part I. Regular algebras, *Fund. Inform.* **2** (1978), 103–127.
58. J. TIURYN, Unique fixed points vs. least fixed points, *Theoret. Comput. Sci.* **12** (1980), 229–254.